Project 3 Retail Marketing

Xuhui Ying

11/15/2022

- Load Libraries
- Load Data & Deal with Missing Values
- Import Data
- Explore Numerics
- Explore Character Variables
- create clusters
- visually choose number of clusters (Elbow Plot)
- build clusters
- explore clusters
- Data Transformation
    - Partition my Data into 70/30 train/test split
    - Recipe
    - Bake
    - Fit Logistic Regression Model
    - Prep for Evaluation
    - Evaluate Logistic Regression Model
- Reduced Model - backward elimination (stepAIC)
    - Use tidymodel framework to fit
    - Evaulate Reduced Regression Model
    - Define Recipe & Bake
    - Define KNN Model
    - KNN Workflow
    - Score KNN model
    - Evaluate (KNN = 7)
    - Define Decision Tree Model
    - Decision Tree Evaluation
    - Decision Tree Evaluate
    - Define Random Forest Model
    - Random Forest Workflow
    - Score Random Forest Model
    - Evaluation (rf_model)
    - Random Forest
    - Final Fit Random Forest
    - Evaluate the Random Forest Model
    - XGBoost Model Buiding
    - Final Fit XGB
    - Evaluate the XGBoost Model

- Prediction

Load Libraries

```
options(warn = -1)
options(scipen = 999) # turns off scientific notation
library(tidyverse)
library(tidymodels)
library(dplyr)
library(janitor)
library(skimr)
library(modelr)
library(GGally)
library(kableExtra) # make nice looking results when we knit
library(vip)        # variable importance
library(fastshap)   # shapley values for variable importance
library(MASS)
library(tree)
library(ggplot2)
library(factoextra)
library(rpart.plot) # plotting decision trees
```

Load Data & Deal with Missing Values

```
retail <- read_csv("marketing_campaign-1.csv") %>% clean_names()

new_customers <- read_csv("new_customers_mkt.csv") %>% clean_names()

head(retail)
```

id <dbl>	birth <dbl>	education <chr>	mar_stat <chr>	income <dbl>	kids <dbl>	teens <dbl>	dt_customer <chr>	recency <dbl>	wines <dbl>
5524	1957	Graduation	Single	58138	0	0	4/9/2012	58	635
2174	1954	Graduation	Single	46344	1	1	8/3/2014	38	11
4141	1965	Graduation	Partner	71613	0	0	21-08-2013	26	426
6182	1984	Graduation	Partner	26646	1	0	10/2/2014	26	11
5324	1981	PhD	Married	58293	1	0	19-01-2014	94	173
7446	1967	Master	Partner	62513	0	1	9/9/2013	16	520

6 rows | 1-10 of 29 columns

```
skim(retail)
```

Data summary

Name	retail
Number of rows	2240
Number of columns	29
_____	
Column type frequency:	
character	3
numeric	26
_____	
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
education	0	1	3	10	0	5	0
mar_stat	0	1	4	7	0	8	0
dt_customer	0	1	8	10	0	663	0

Variable type: numeric

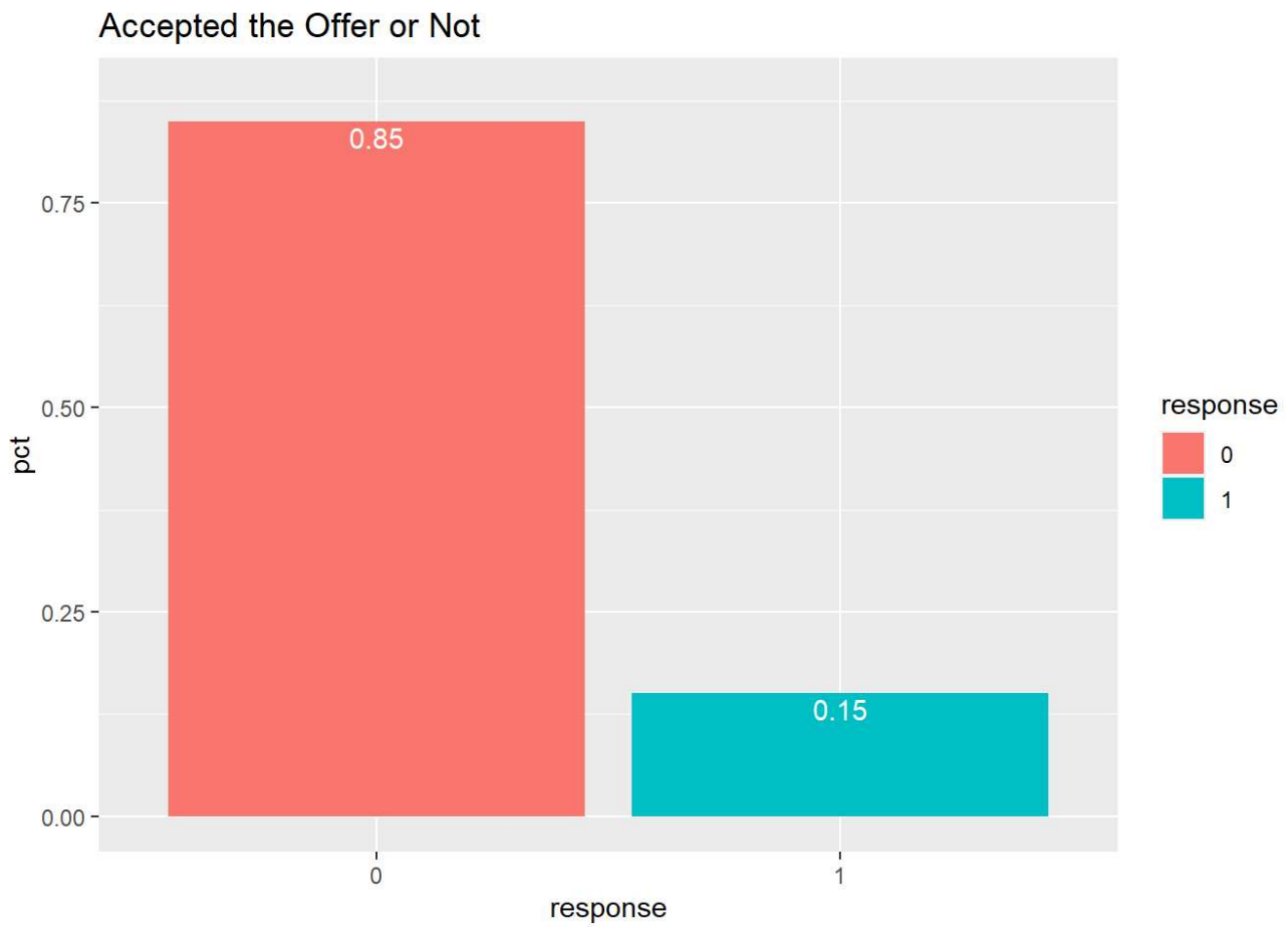
skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
id	0	1.00	5592.16	3246.66	0	2828.25	5458.5	8427.75	11191	▇▇▇▇▇
birth	0	1.00	1968.81	11.981893	1959.00	1970.0	1977.00	1996	▁▂▅▇	
income	24	0.9952247.2525173.08173035303.0051381.568522.00666666								▇▁▁▁▁
kids	0	1.00	0.44	0.54	0	0.00	0.0	1.00	2	▇▁▆▁
teens	0	1.00	0.51	0.54	0	0.00	0.0	1.00	2	▇▁▆▁
recency	0	1.00	49.11	28.96	0	24.00	49.0	74.00	99	▇▇▇▇▇
wines	0	1.00	303.94	336.60	0	23.75	173.5	504.25	1493	▇▂▂▁▁
fruits	0	1.00	26.30	39.77	0	1.00	8.0	33.00	199	▇▁▁▁▁
meat	0	1.00	166.95	225.72	0	16.00	67.0	232.00	1725	▇▁▁▁▁
fish	0	1.00	37.53	54.63	0	3.00	12.0	50.00	259	▇▁▁▁▁
sweets	0	1.00	27.06	41.28	0	1.00	8.0	33.00	263	▇▁▁▁▁
gold	0	1.00	44.02	52.17	0	9.00	24.0	56.00	362	▇▁▁▁▁
deals	0	1.00	2.33	1.93	0	1.00	2.0	3.00	15	▇▃▁▁▁
web	0	1.00	4.08	2.78	0	2.00	4.0	6.00	27	▇▅▁▁▁
catalog	0	1.00	2.66	2.92	0	0.00	2.0	4.00	28	▇▃▁▁▁
store	0	1.00	5.79	3.25	0	3.00	5.0	8.00	13	▃▇▃▅▂
visits	0	1.00	5.32	2.43	0	3.00	6.0	7.00	20	▇▆▁▁▁

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
cmp3	0	1.00	0.07	0.26	0	0.00	0.0	0.00	1	▇▁▁▁▁
cmp4	0	1.00	0.07	0.26	0	0.00	0.0	0.00	1	▇▁▁▁▁
cmp5	0	1.00	0.07	0.26	0	0.00	0.0	0.00	1	▇▁▁▁▁
cmp1	0	1.00	0.06	0.25	0	0.00	0.0	0.00	1	▇▁▁▁▁
cmp2	0	1.00	0.01	0.11	0	0.00	0.0	0.00	1	▇▁▁▁▁
cmplain	0	1.00	0.01	0.10	0	0.00	0.0	0.00	1	▇▁▁▁▁
z_cost	0	1.00	3.00	0.00	3	3.00	3.0	3.00	3	▁▇▁▁▁
z_rev	0	1.00	11.00	0.00	11	11.00	11.0	11.00	11	▁▇▁▁▁
response	0	1.00	0.15	0.36	0	0.00	0.0	0.00	1	▇▁▁▁▂

```
retail$income[is.na(retail$income)]<-median(retail$income,na.rm=TRUE)
```

Import Data

```
retail$response <- as.factor(retail$response)

retail_summary <- retail %>%
  count(response) %>%
  mutate(pct = n/sum(n))

retail_summary %>%
  ggplot(aes(x=response, y=pct, fill=response)) +
  geom_col() +
  geom_text(aes(x=factor(response),y=pct+0.034, label = round(pct,2)), vjust = 2.5, colour = "white") +
  labs(title = "Accepted the Offer or Not")
```




```
retail %>%
  group_by(response) %>%
  summarize(n=n()) %>%
  ungroup() %>%
  mutate(pct = n/sum(n))
```

response	n	pct
<fct>	<int>	<dbl>
0	1904	0.85
1	336	0.15

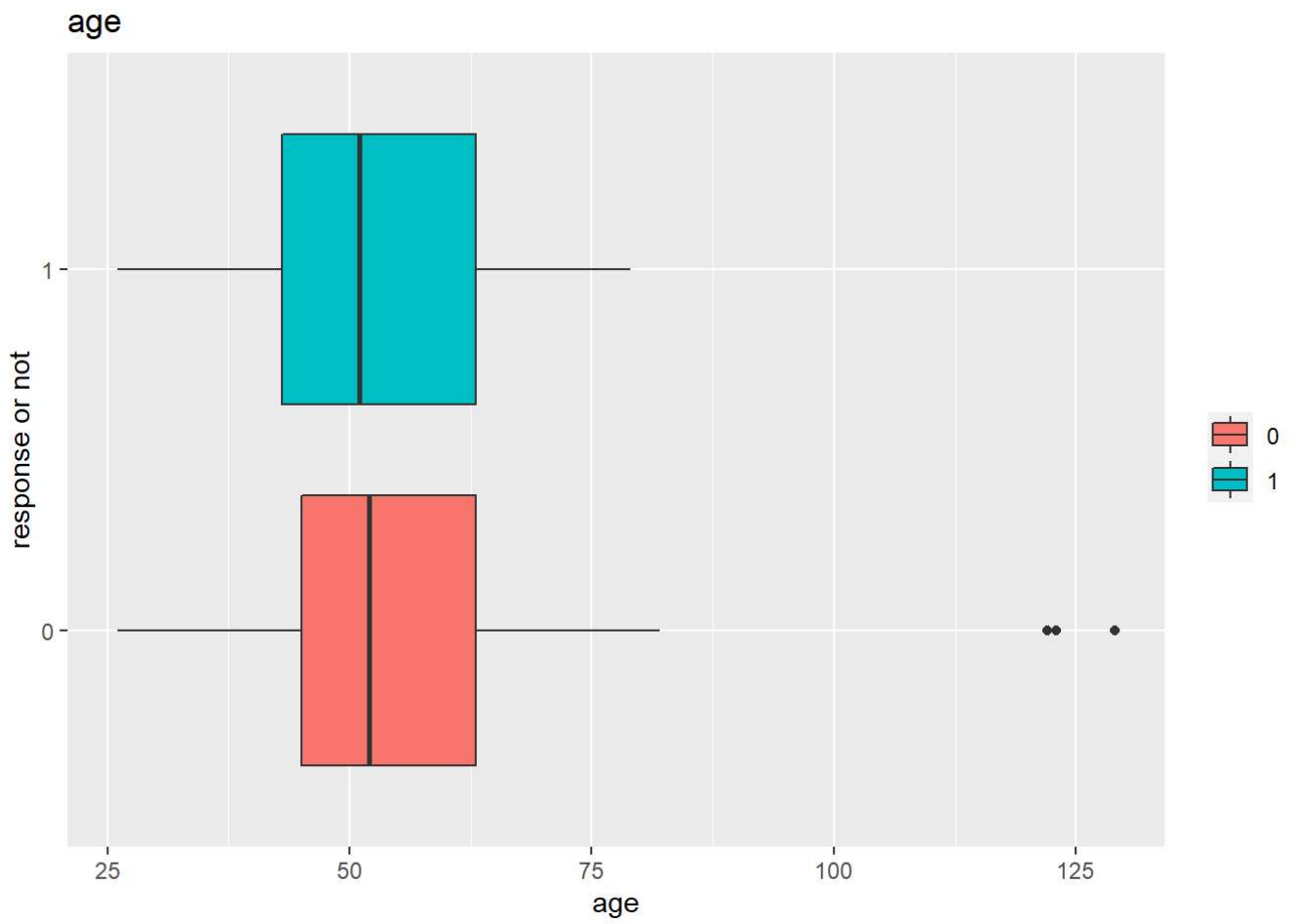
2 rows

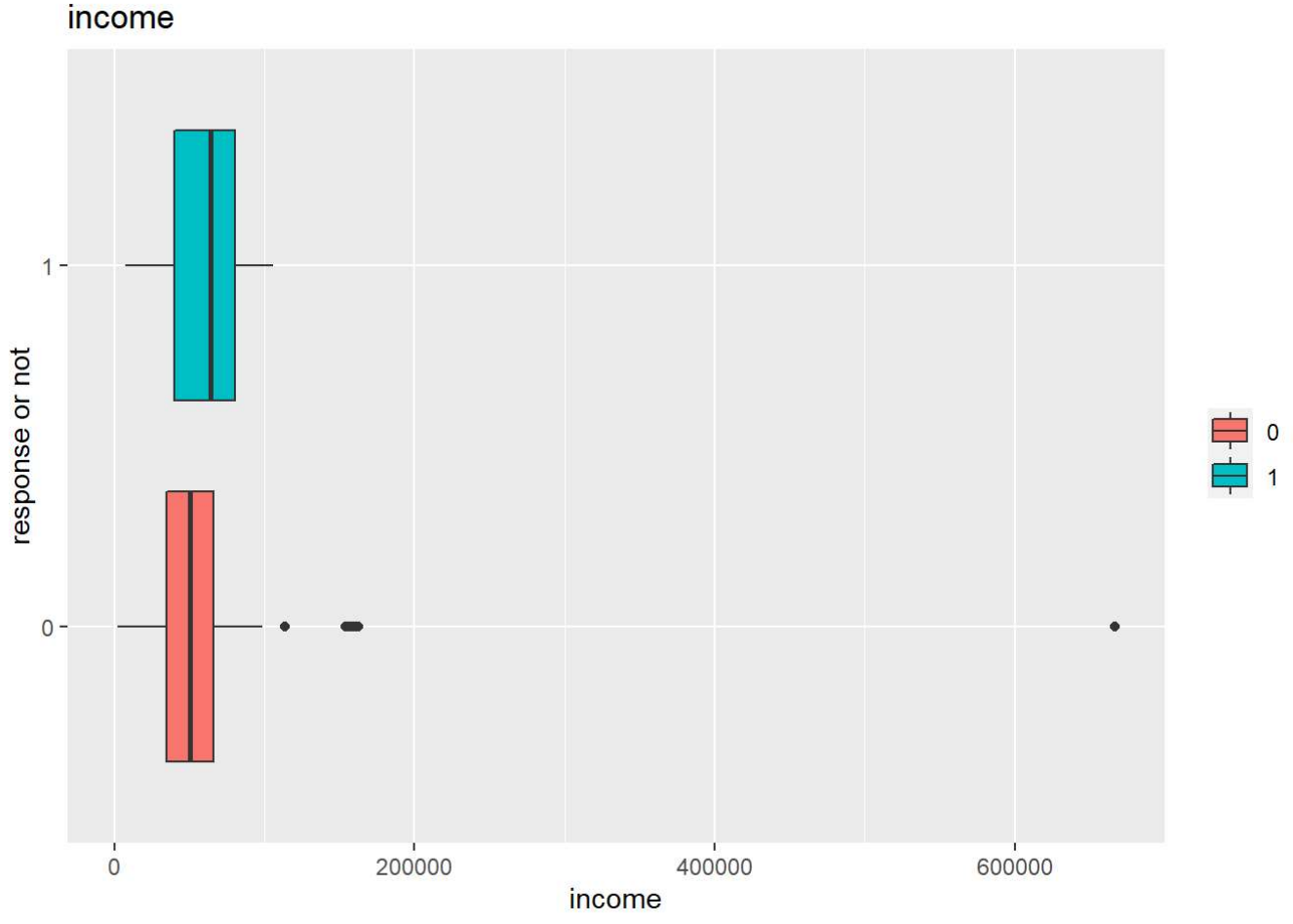
```
retail <- retail %>% mutate(cmp1 = as.character(cmp1))
retail <- retail %>% mutate(cmp2 = as.character(cmp2))
retail <- retail %>% mutate(cmp3 = as.character(cmp2))
retail <- retail %>% mutate(cmp4 = as.character(cmp2))
retail <- retail %>% mutate(cmp5 = as.character(cmp2))
retail <- retail %>% mutate(cmplain = as.character(cmplain))
```

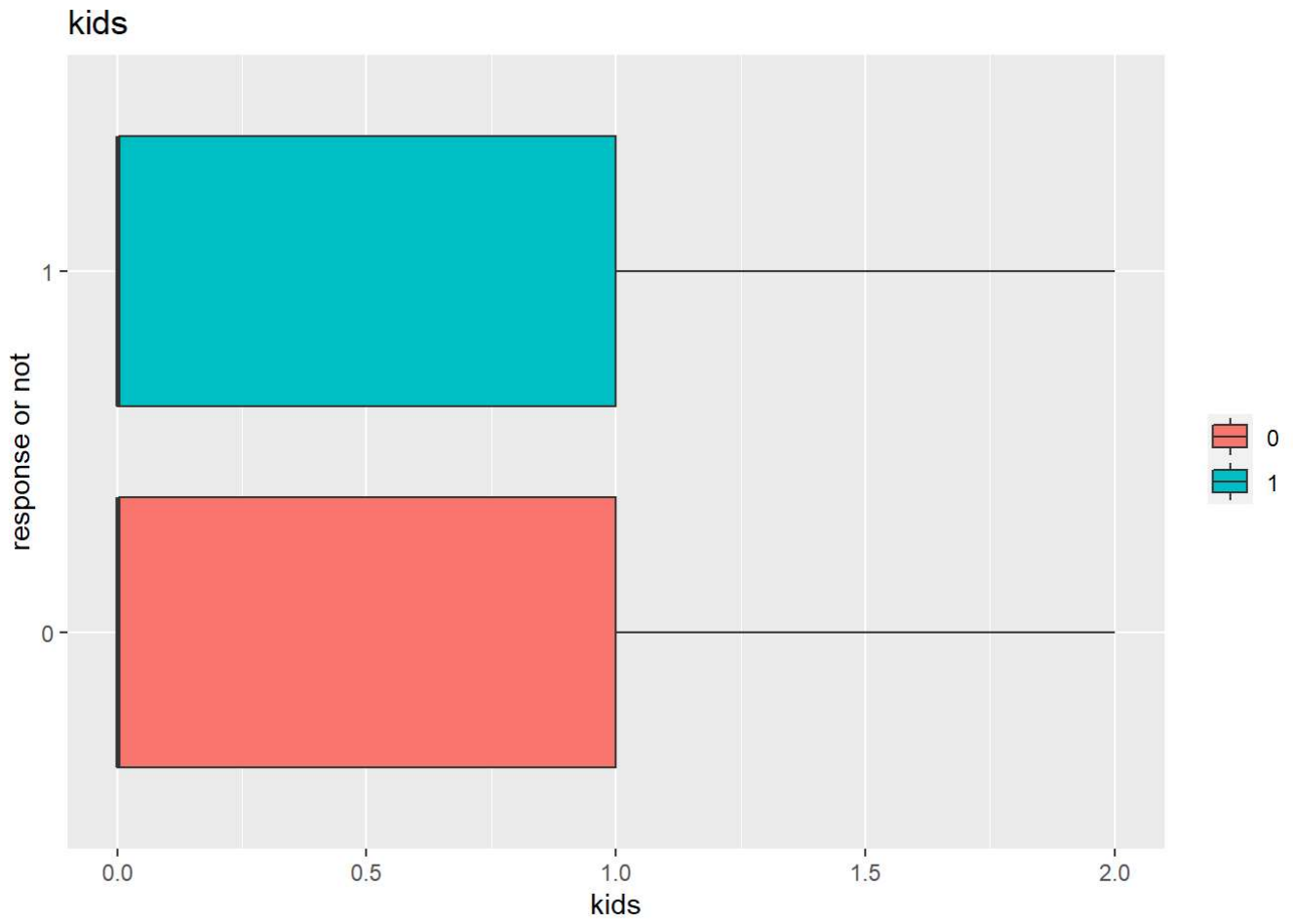
Explore Numerics

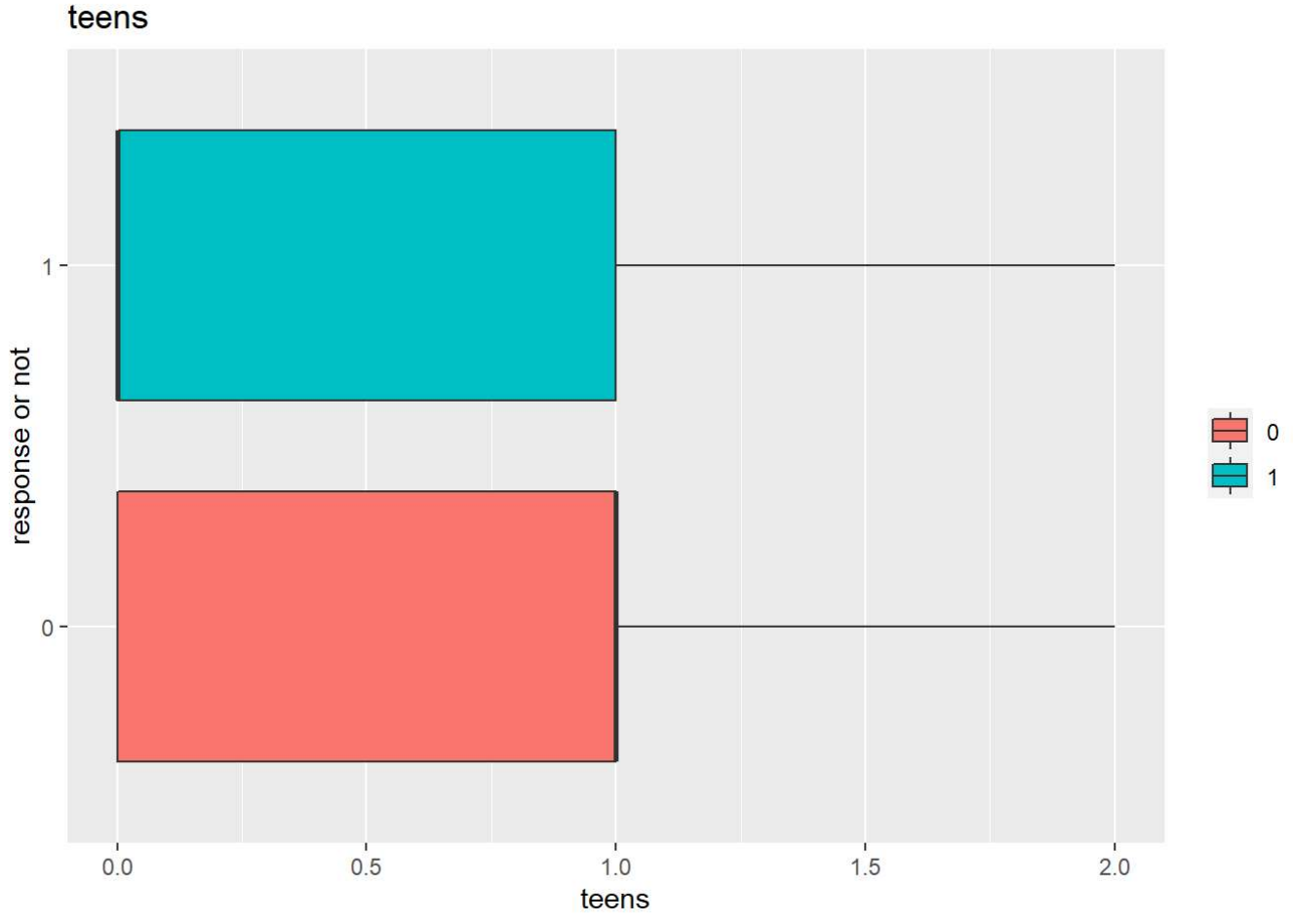
numeric variables: birth, income, kids, teens, recency, wines, fruits, meat, fish, sweets, gold, deals, web, catalog, store, visits, z_cost, z_rev

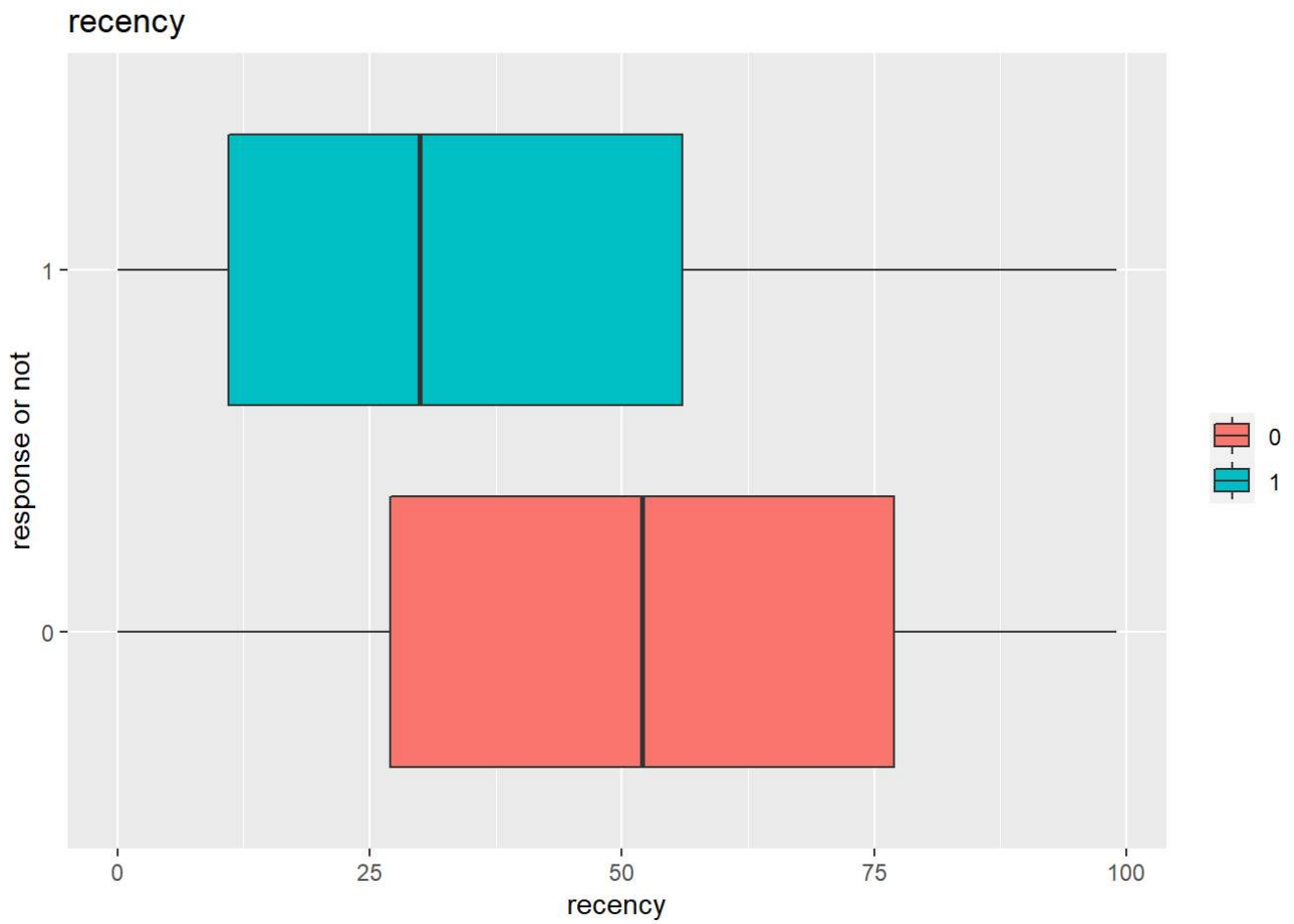
```
retail_numerics <- retail %>% mutate(age = 2022 - birth)


# -- comparative boxplots


boxplot <- function(m){
    ggplot(retail_numerics, aes(x=!!as.name(m), y=as.factor(response), fill=as.factor(response))) +
    geom_boxplot() +
    labs(title = as.character(m), y = 'response or not') +
    theme(legend.title = element_blank())
}

numerics <- c('age', 'income', 'kids', 'teens', 'recency', 'wines', 'fruits', 'meat', 'fish', 'sweets', 'gol
         d', 'deals', 'web', 'catalog', 'store', 'visits', 'z_cost', 'z_rev')


for (c in numerics){
    print(boxplot(c))
}
```

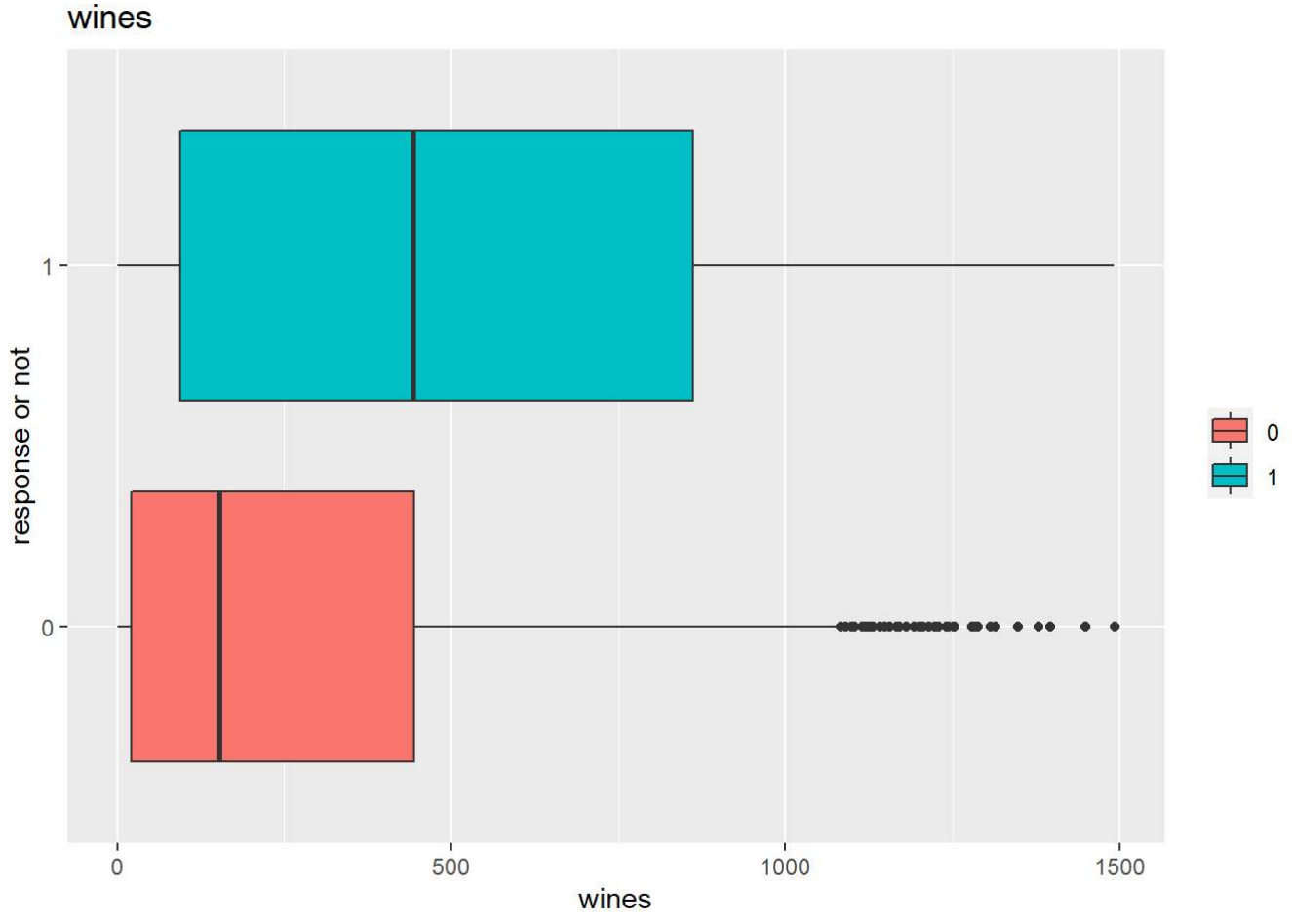

age
response or not
1
0
25
50
75
100
125
age
0
1

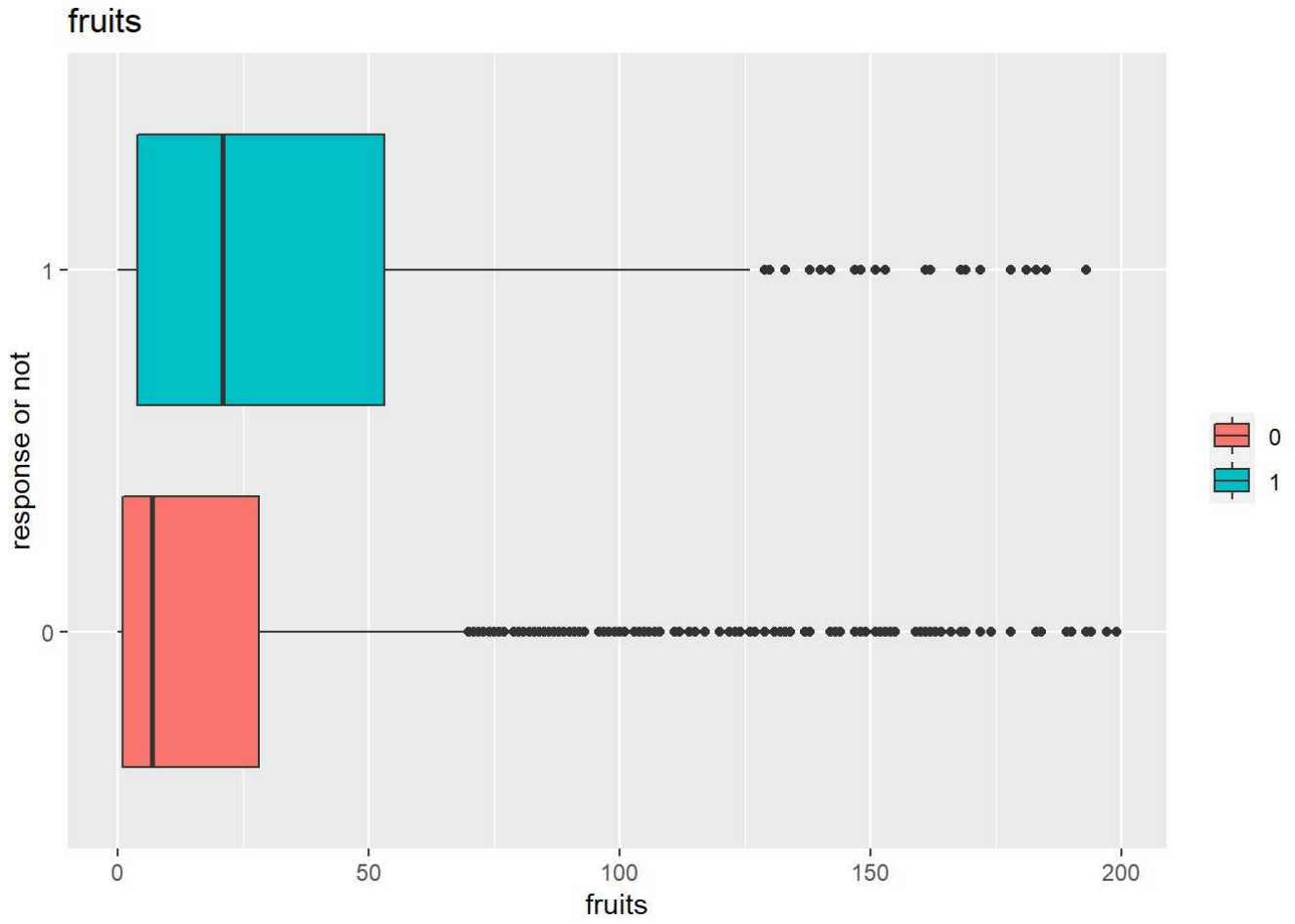

income
response or not
1
0
0
200000
400000
600000
income
0
1

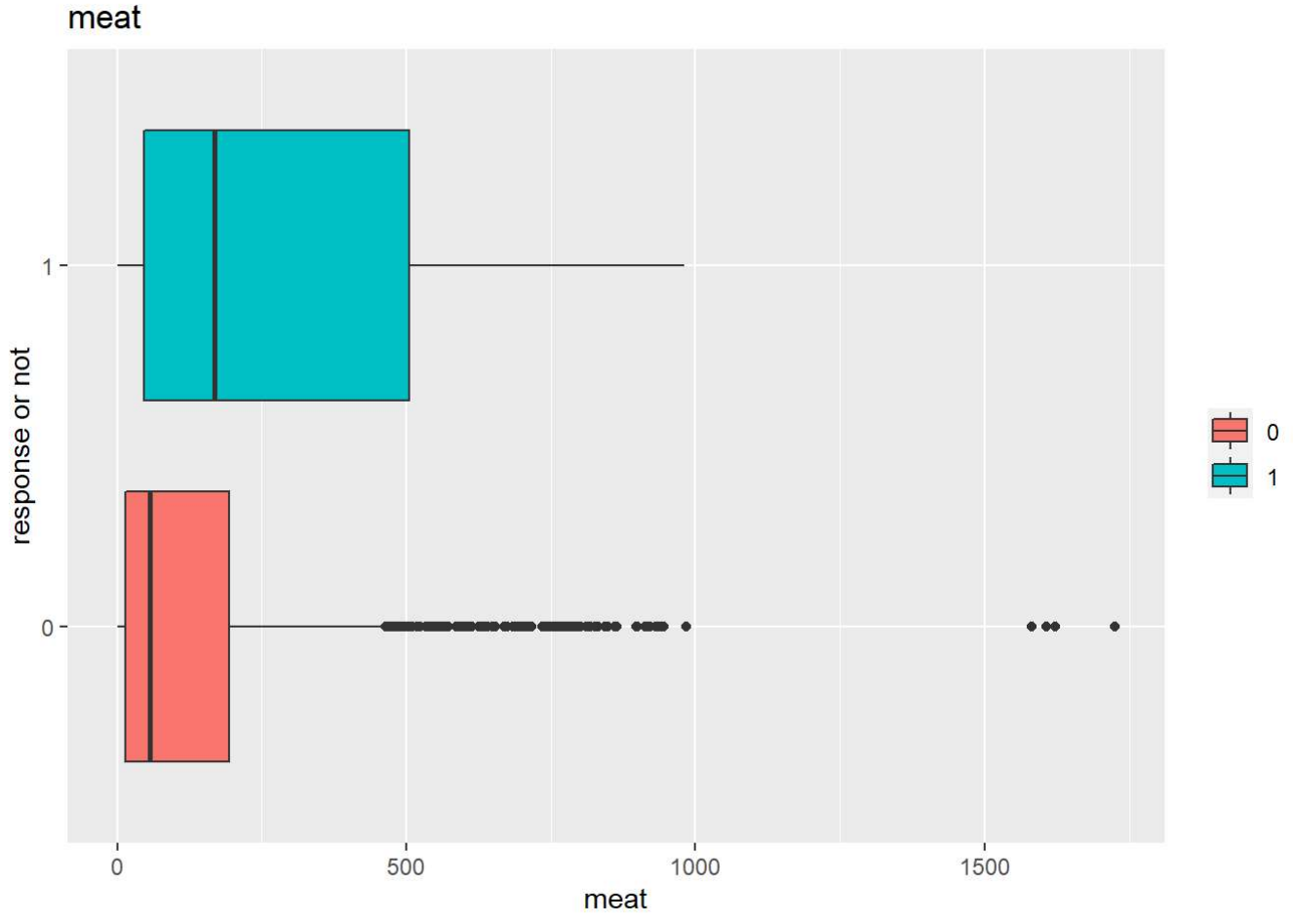

kids
response or not
1
0
0.0
0.5
1.0
1.5
2.0
kids
0
1

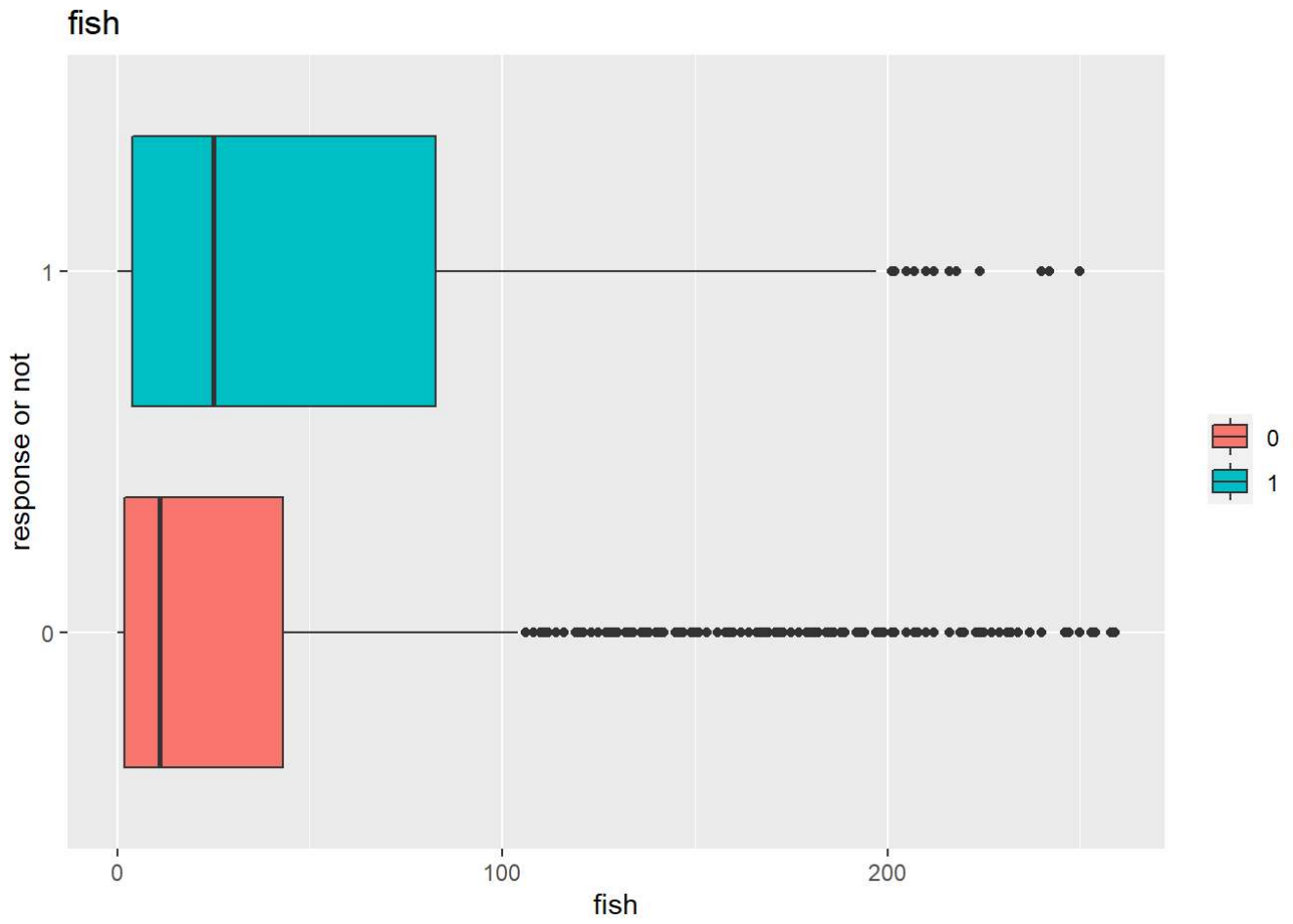

teens
response or not
1
0
0.0
0.5
1.0
1.5
2.0
teens
0
1

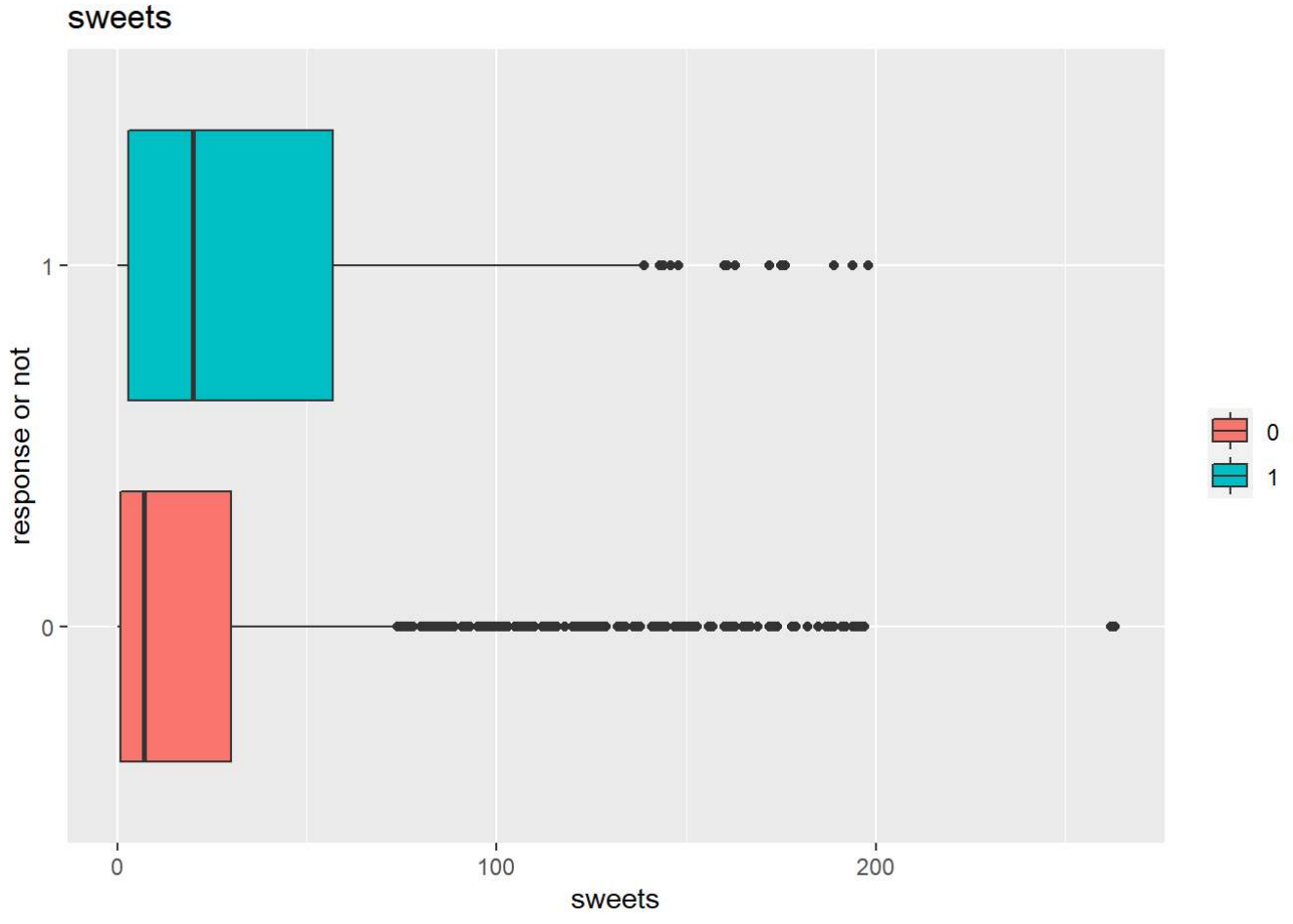

recency
response or not
1
0
0
25
50
75
100
recency
0
1

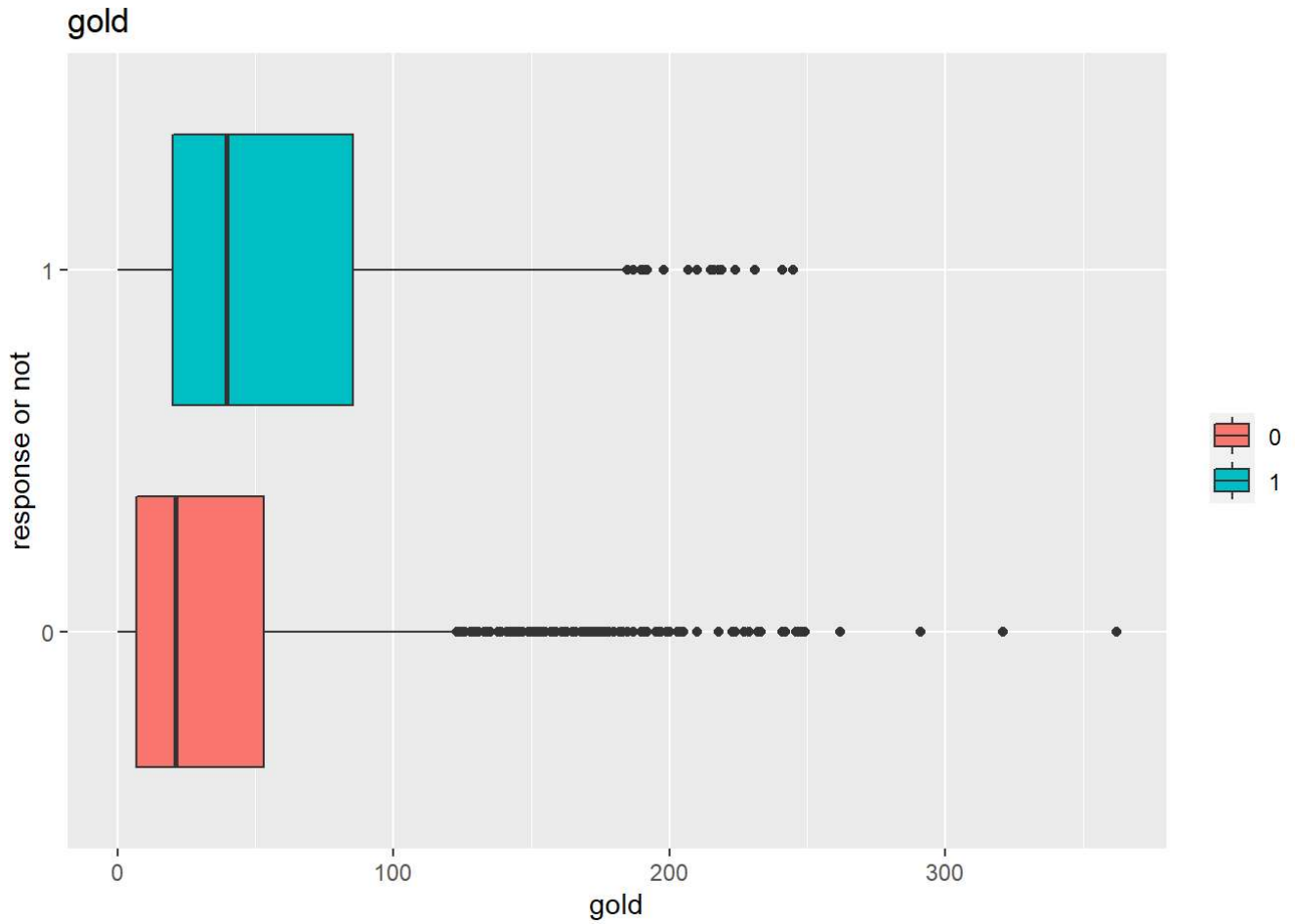

wines
response or not
1
0
0
500
1000
1500
wines
0
1

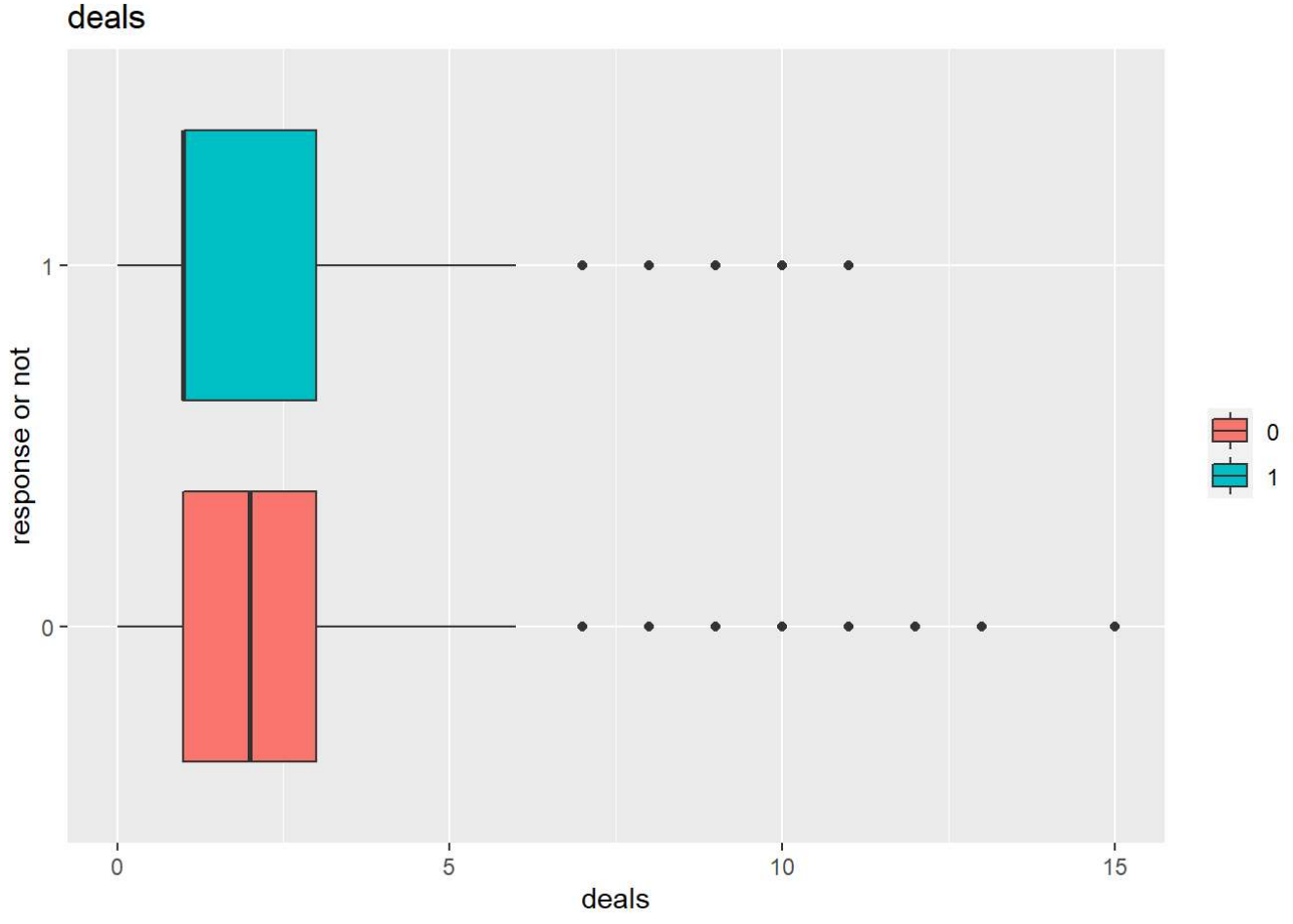

fruits
response or not
1
0
0
50
100
150
200
fruits
0
1

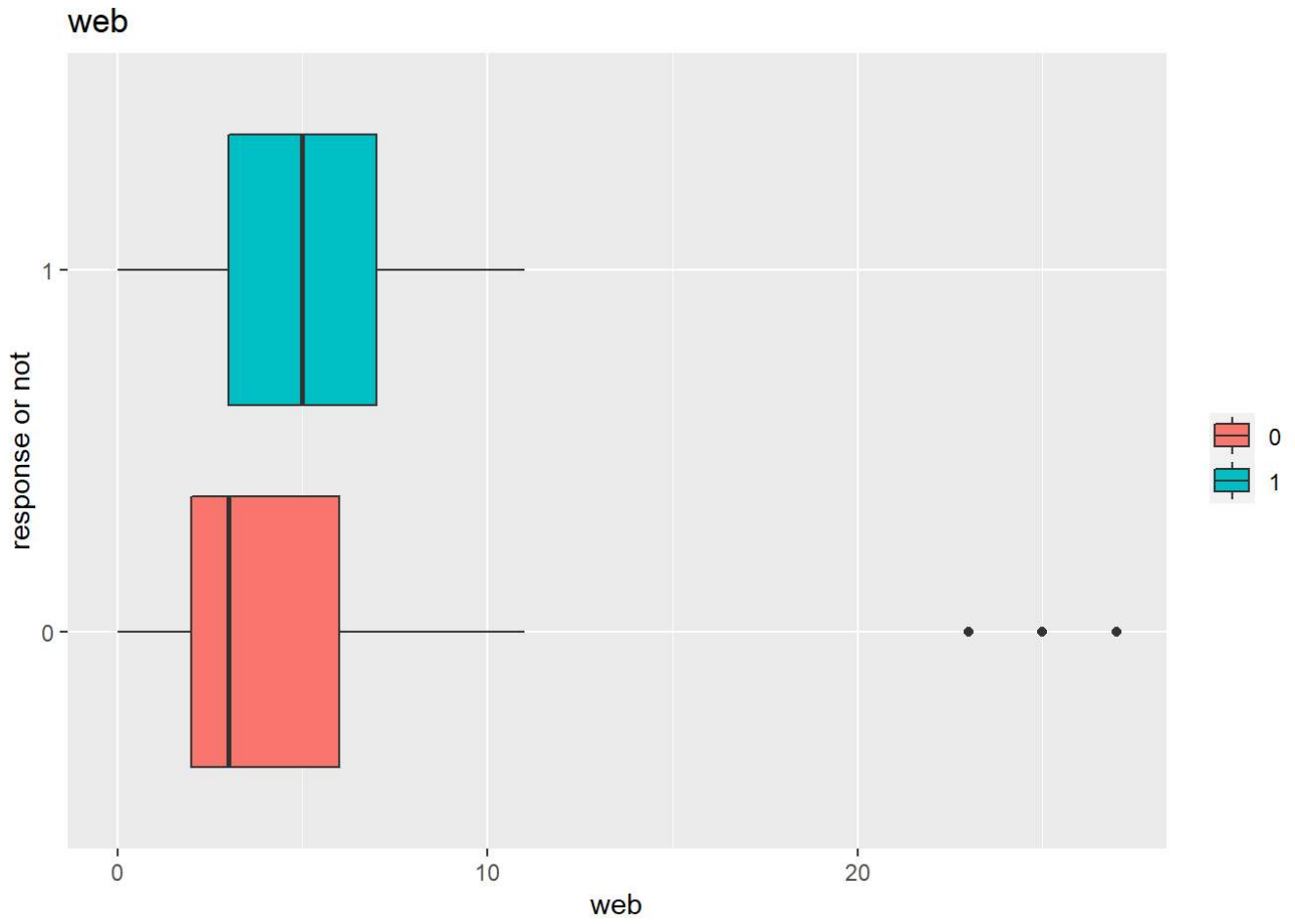

meat
response or not
1
0
0
500
1000
1500
meat
0
1

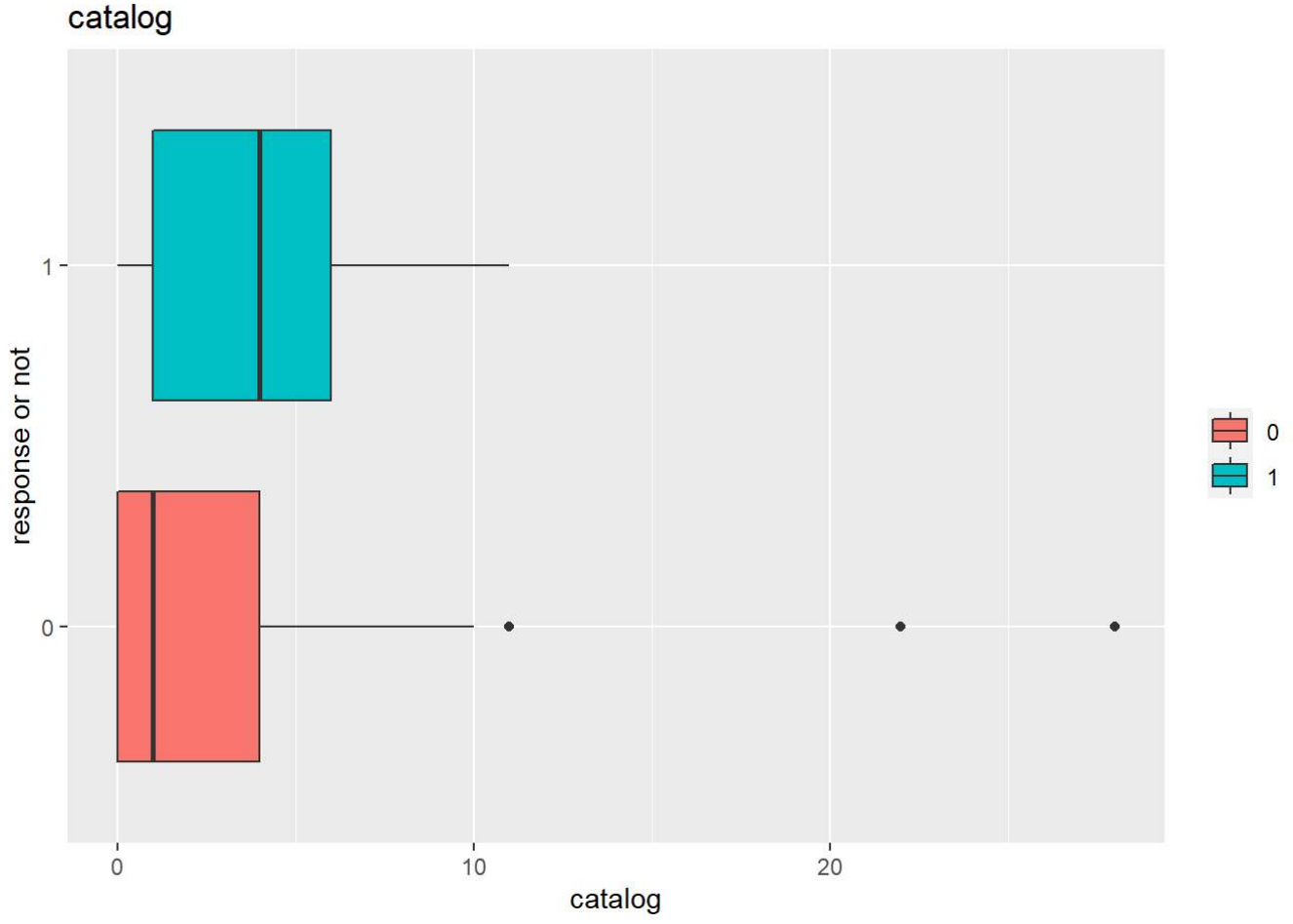

fish
response or not
1
0
0
100
200
fish
0
1

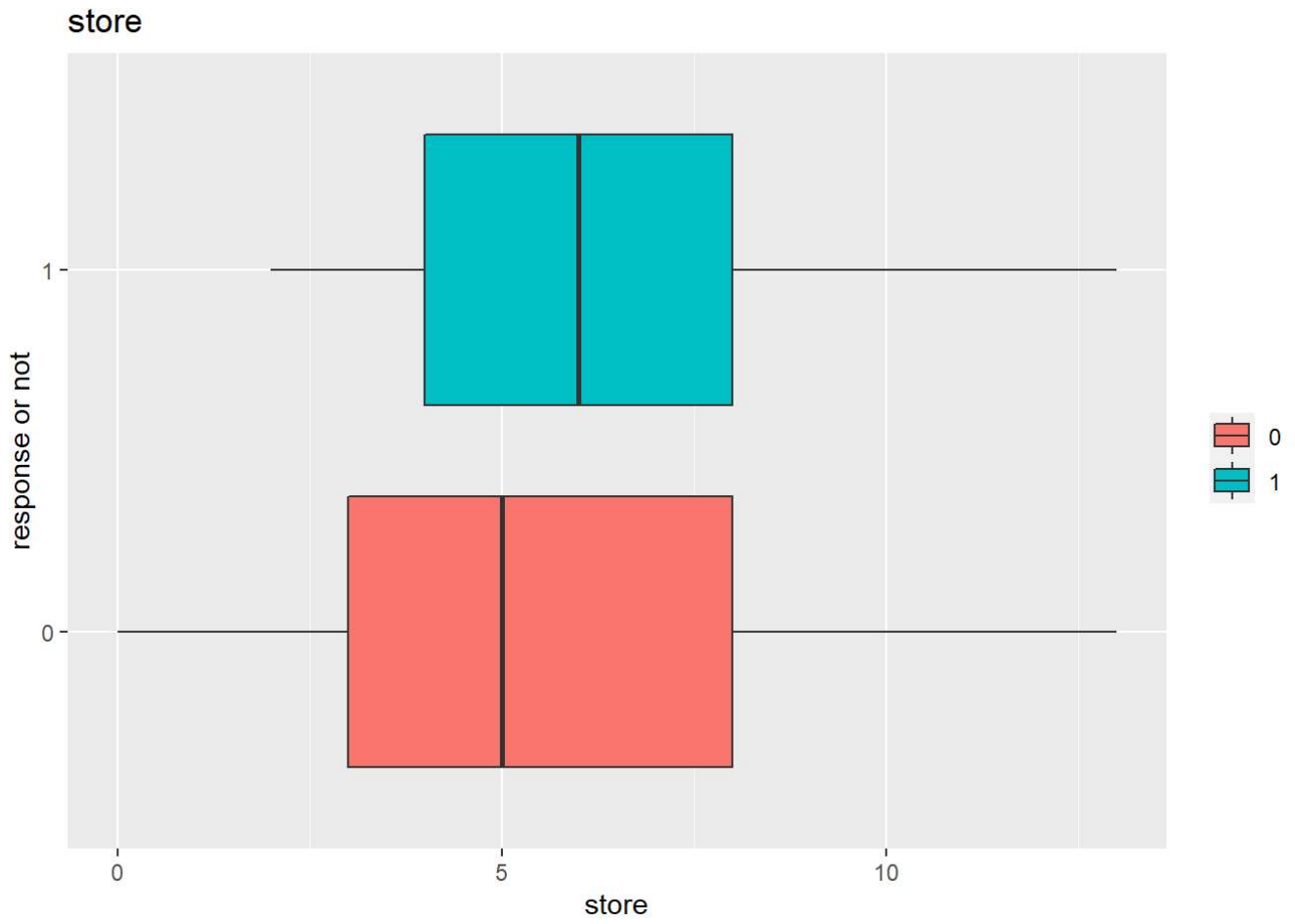

sweets
response or not
1
0
0
100
200
sweets
0
1

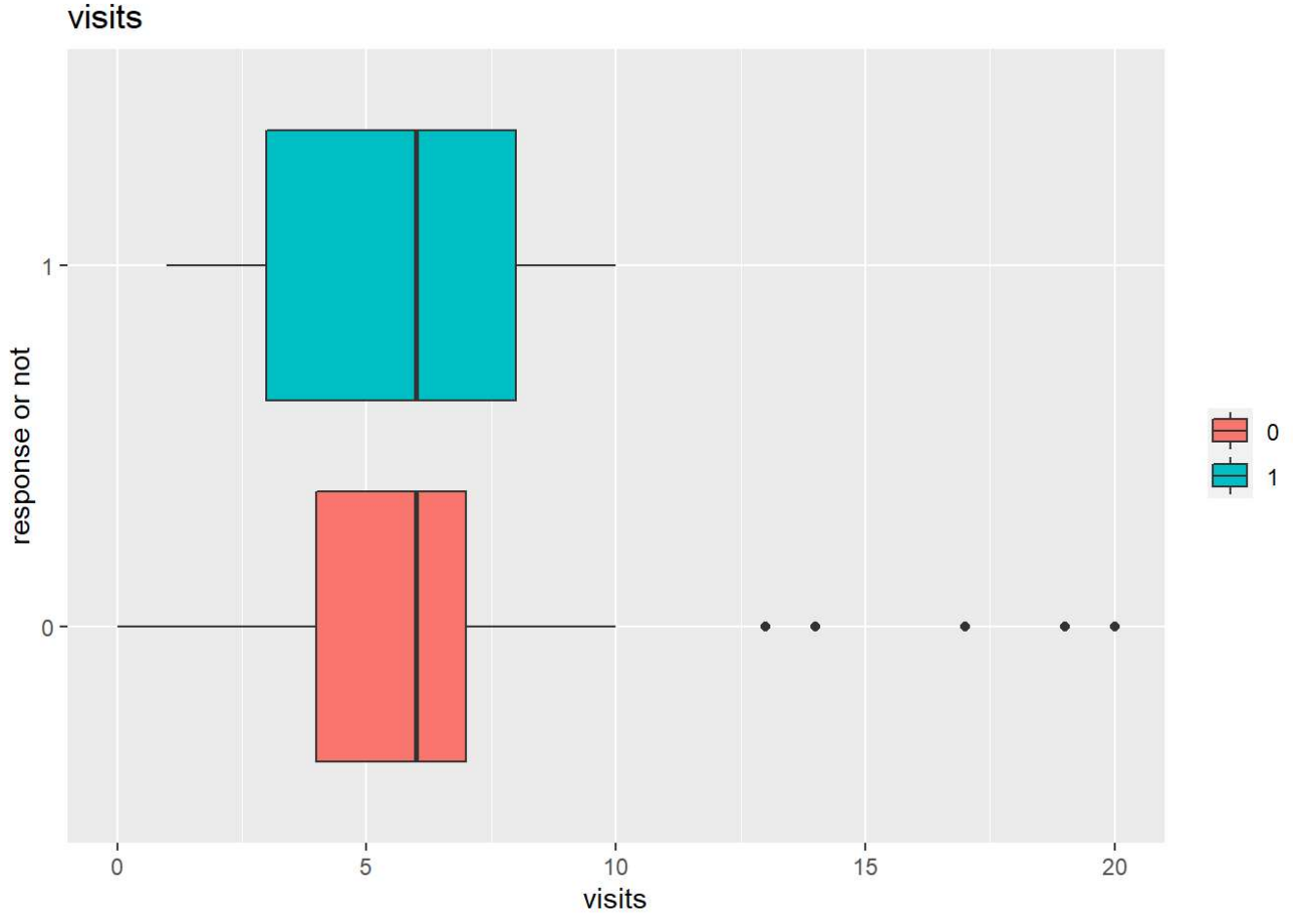

gold
response or not
1
0
0
100
200
300
gold
0
1

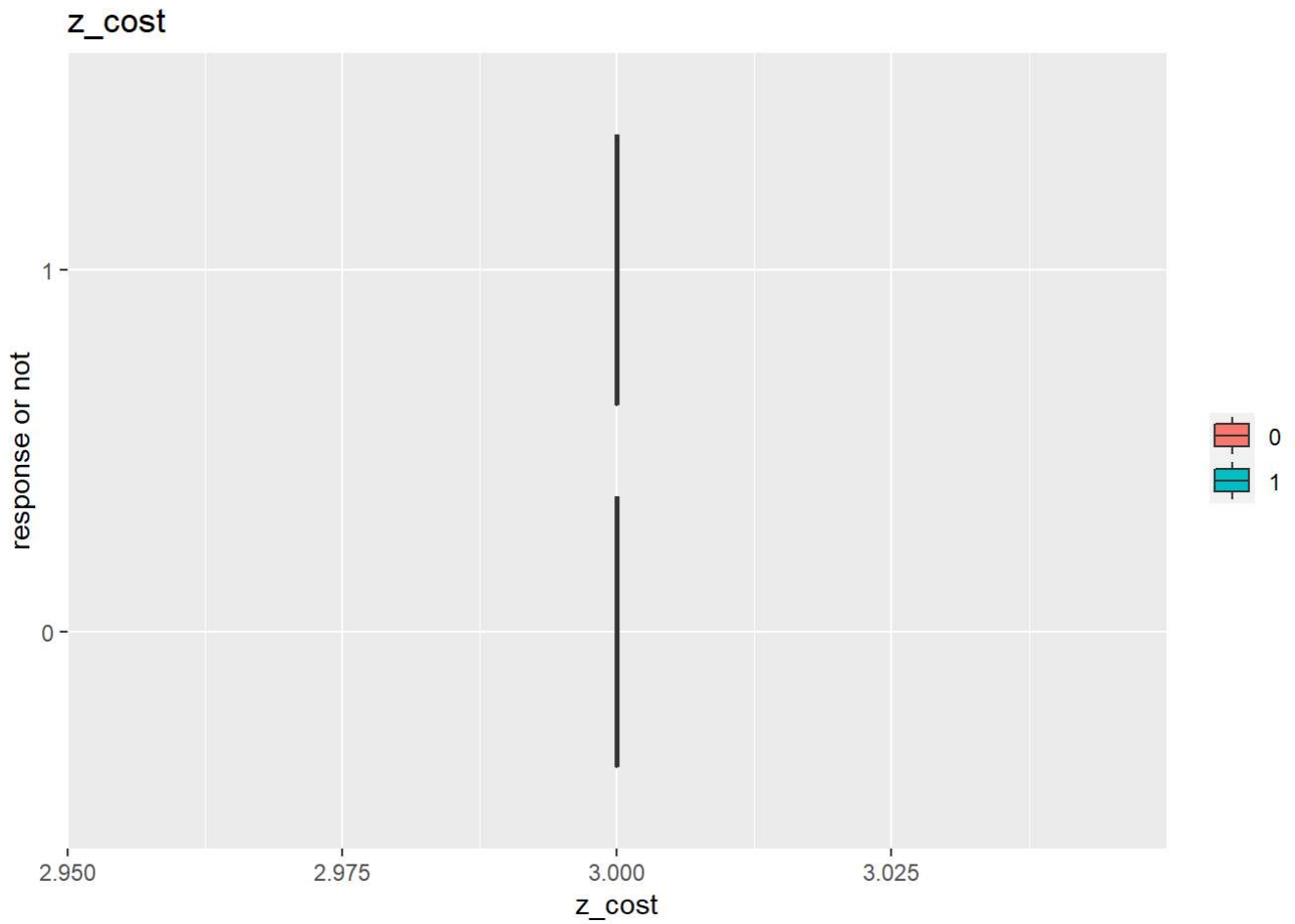

deals
response or not
1
0
0
5
10
15
deals
0
1

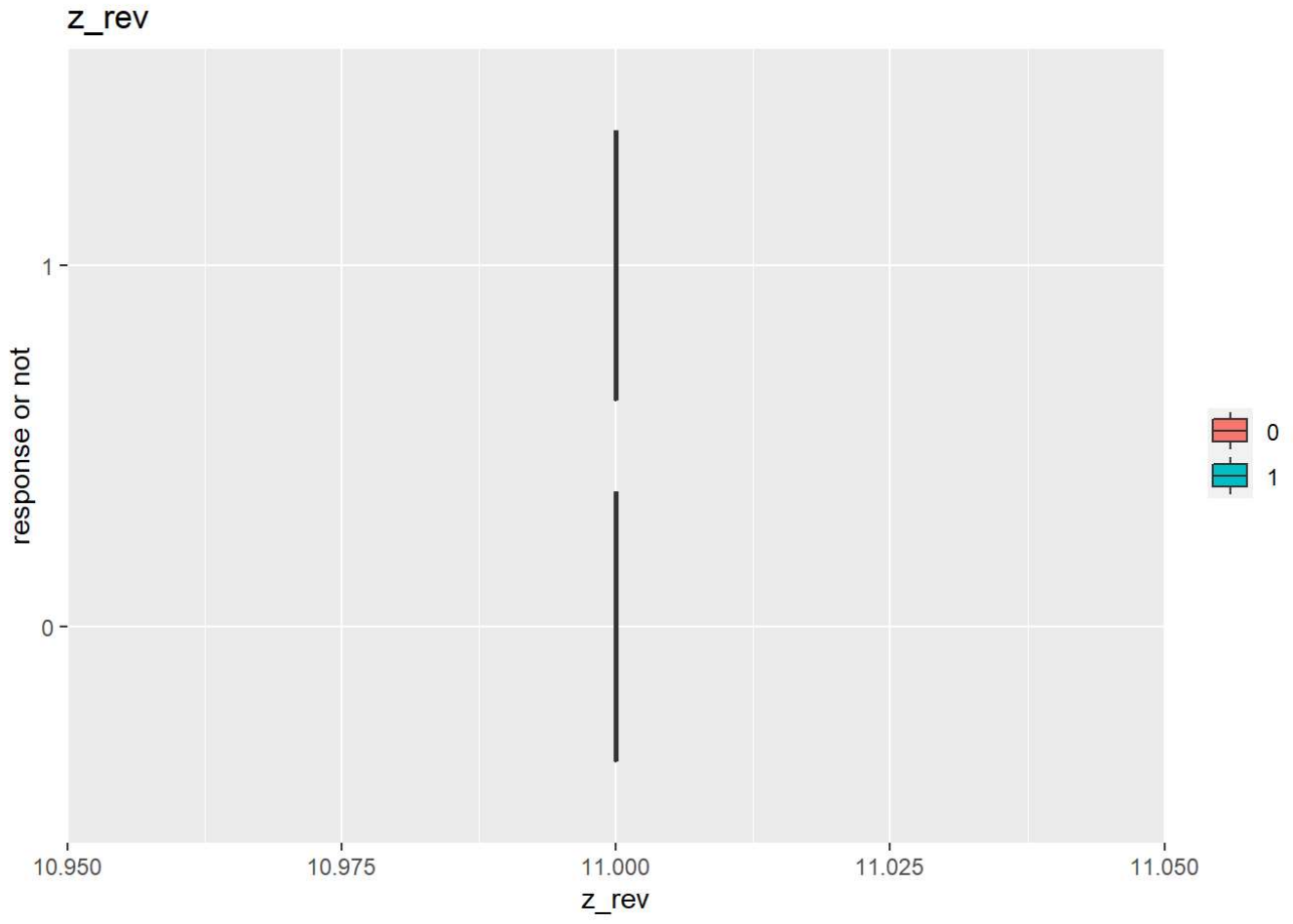

web
response or not
1
0
0
10
20
web
0
1


catalog
response or not
1
0
0
10
20
catalog
0
1


store
response or not
1
0
0
5
10
store
0
1


visits
response or not
1
0
0
5
10
15
20
visits
0
1

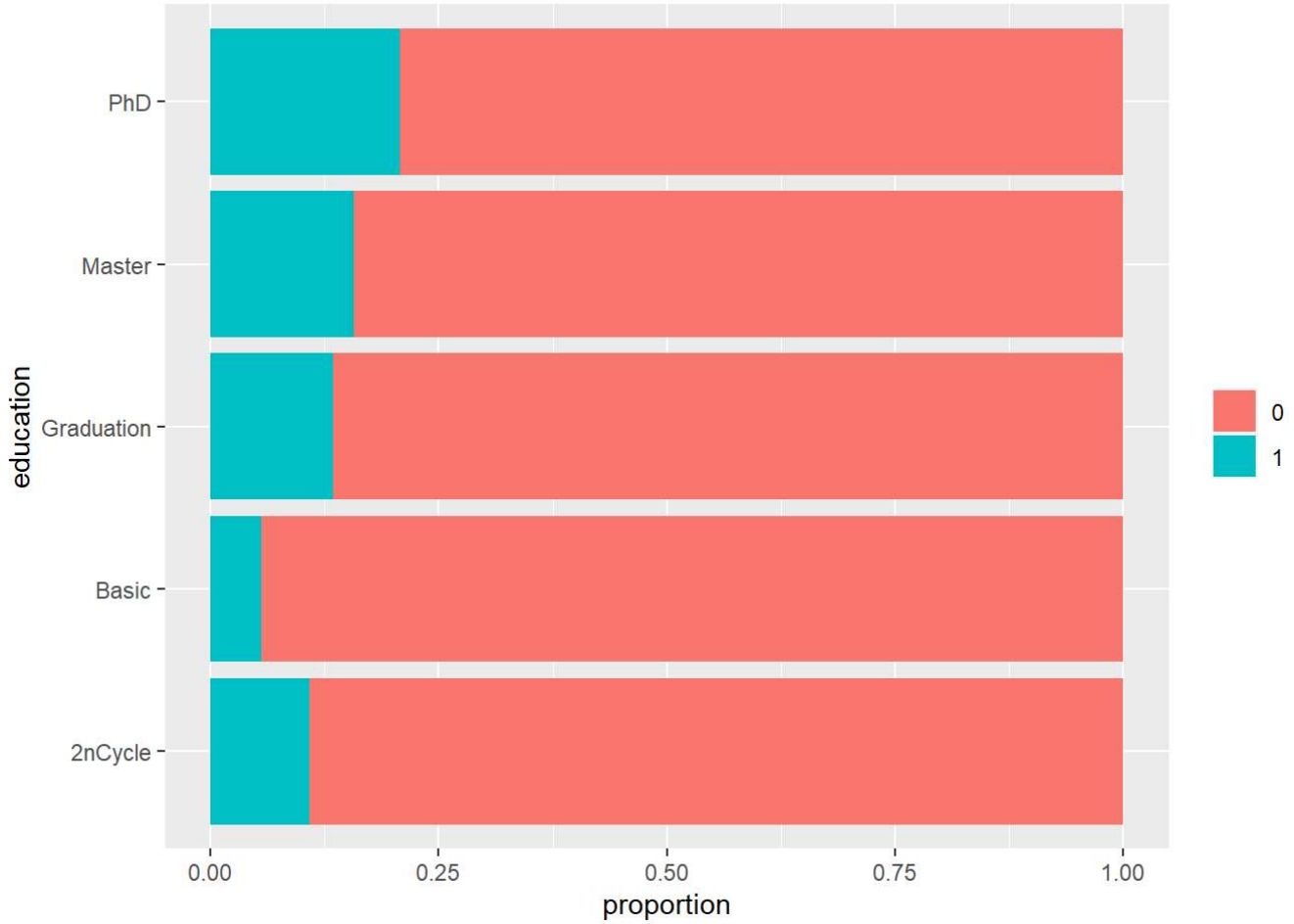

z_cost
response or not
1
0
2.950
2.975
3.000
3.025
z_cost
0
1

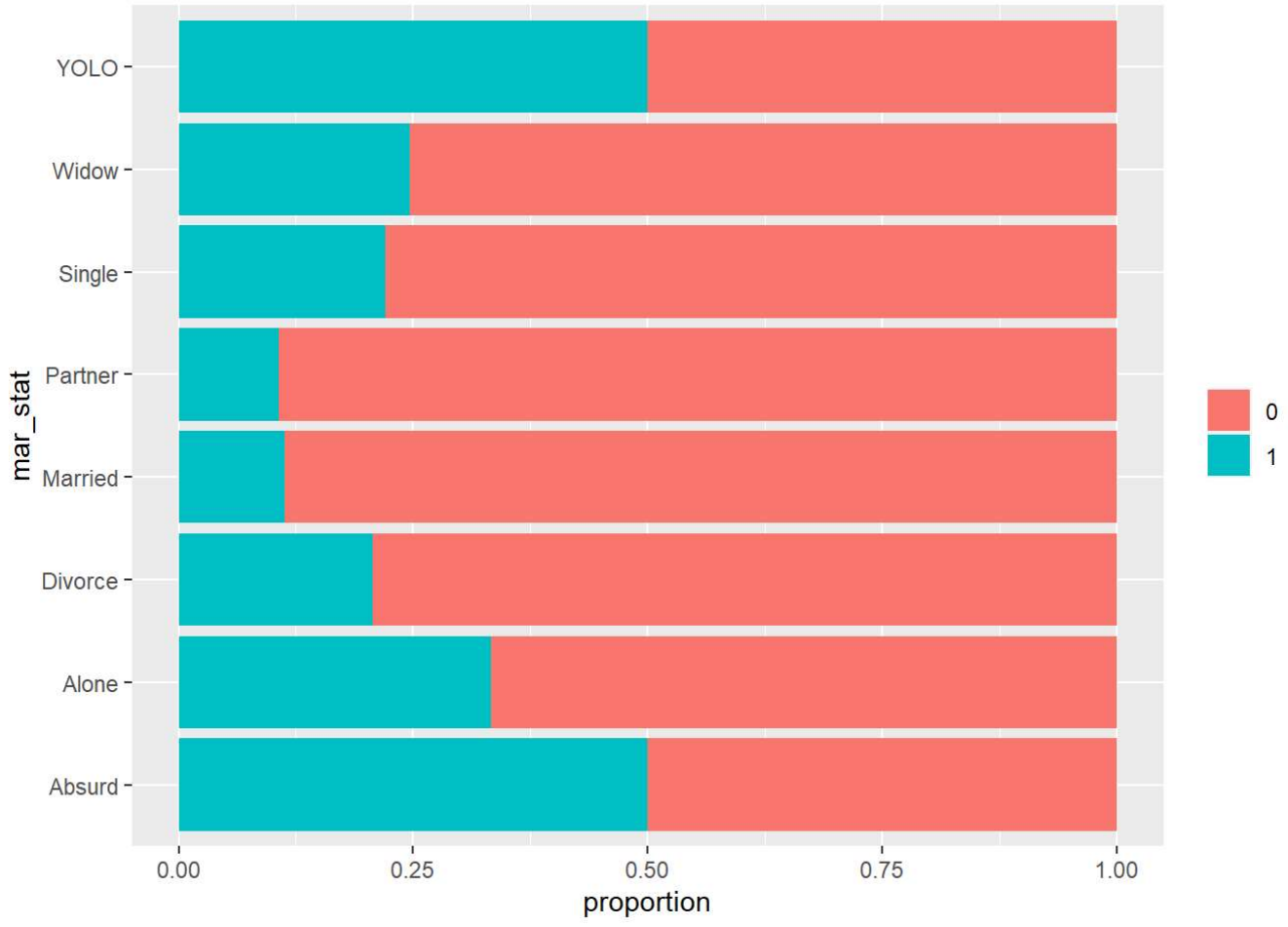

z_rev
response or not
1
0
10.950
10.975
11.000
11.025
11.050
z_rev
0
1

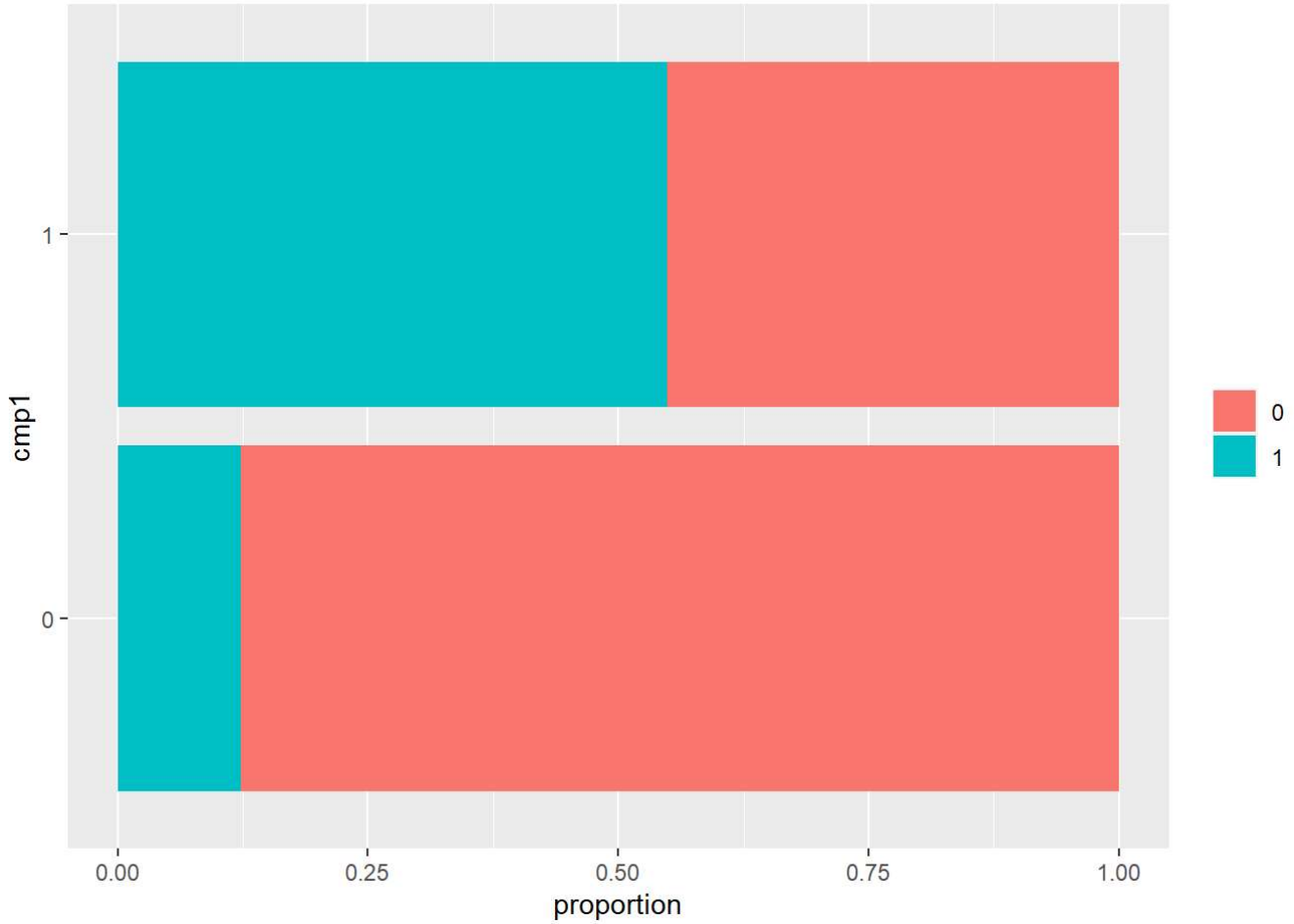
Explore Character Variables

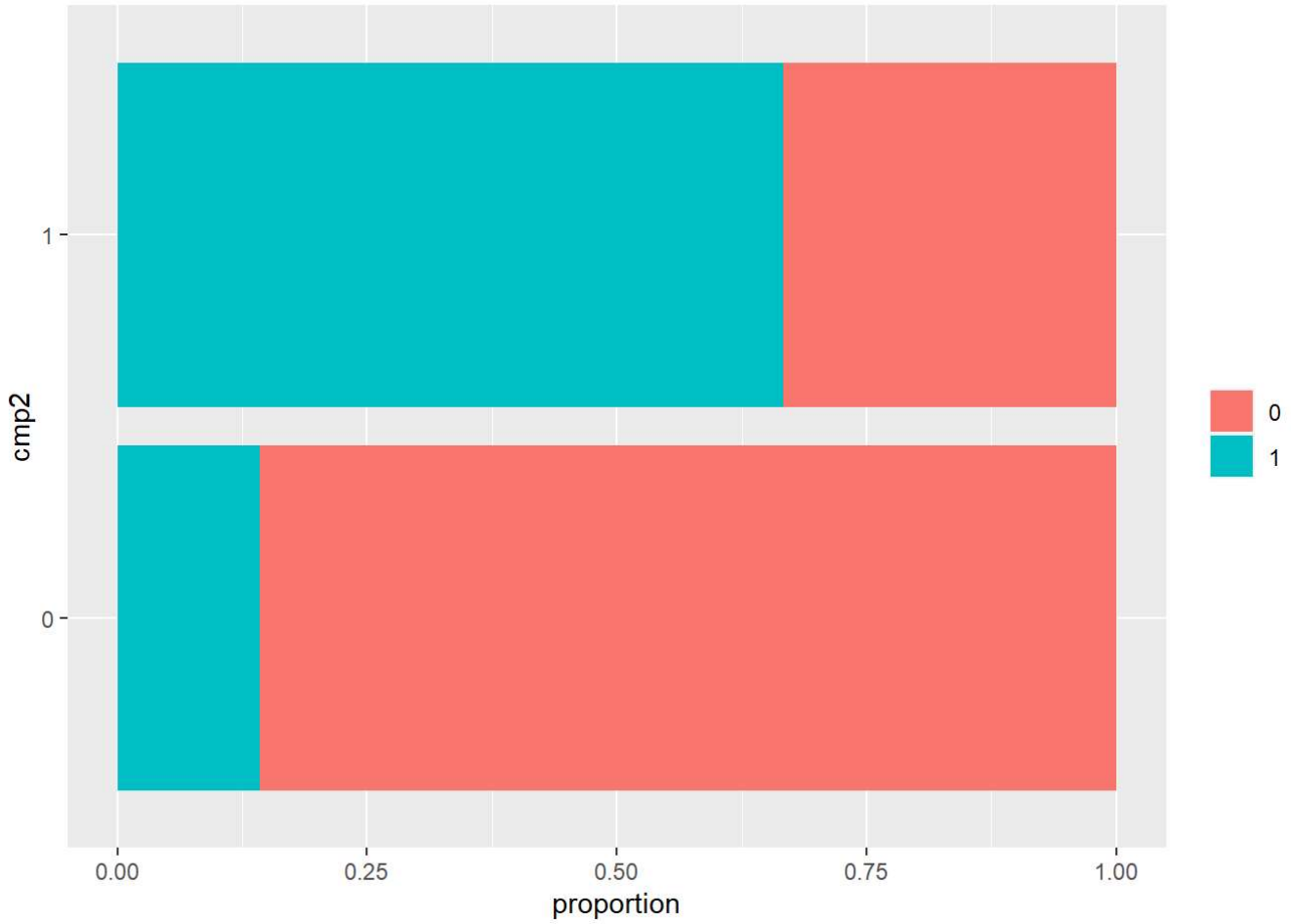
categorical variables: education, mar_stat, cmp1, cmp2, cmp3, cmp4, cmp5, cmplain

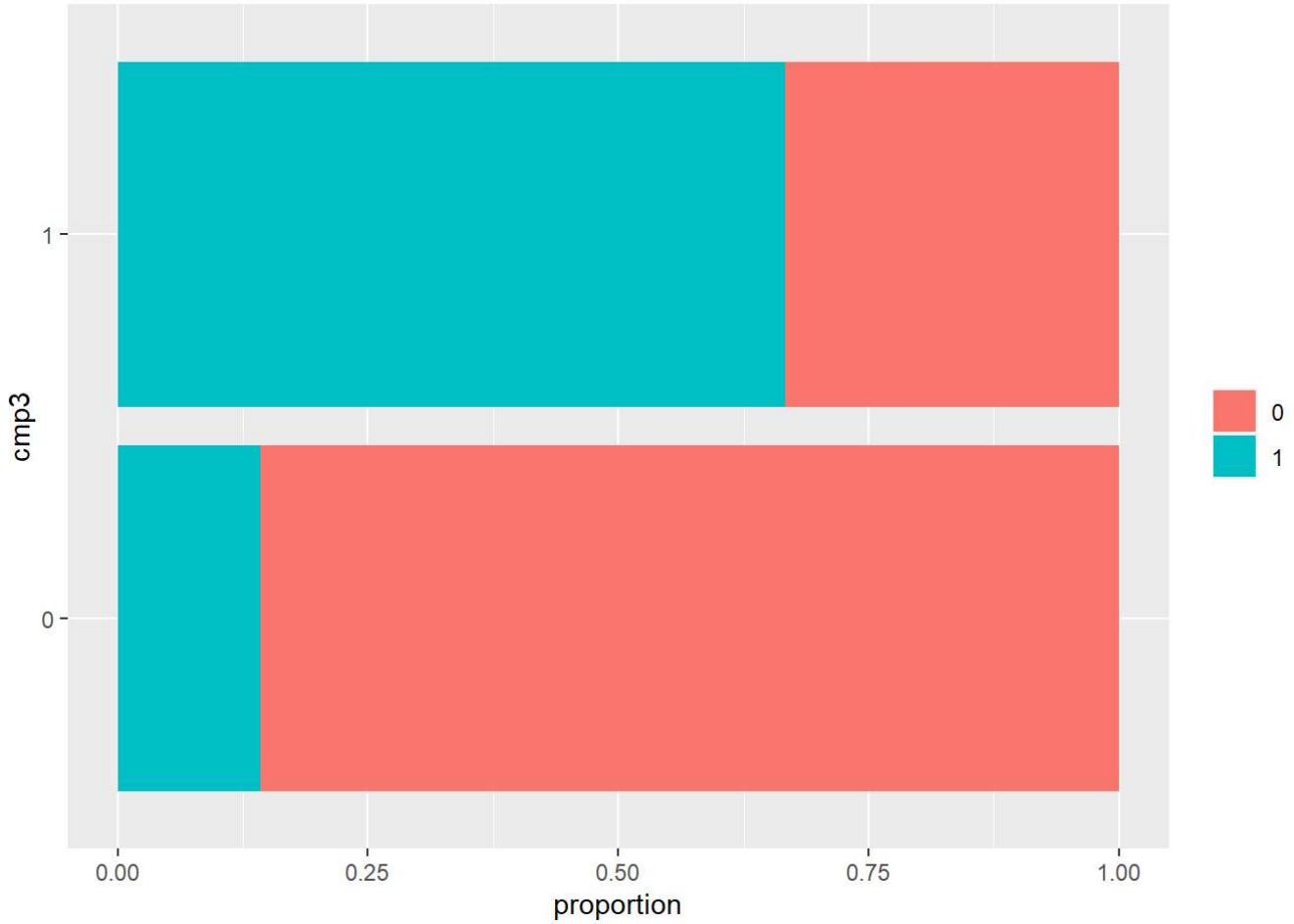
```
char_fill <- function(col){
    retail %>%
    na.omit() %>%
    ggplot(aes(!!as.name(col), fill = as.factor(response))) +
    geom_bar(position = 'fill') +
    coord_flip() +
    labs(y = 'proportion') +
    theme(legend.title = element_blank())
}

dummy <- c('education', 'mar_stat', 'cmp1', 'cmp2', 'cmp3', 'cmp4', 'cmp5', 'cmplain')

# -- for each character column, create a chart
for (column in dummy){
    print(char_fill(column))
}
```

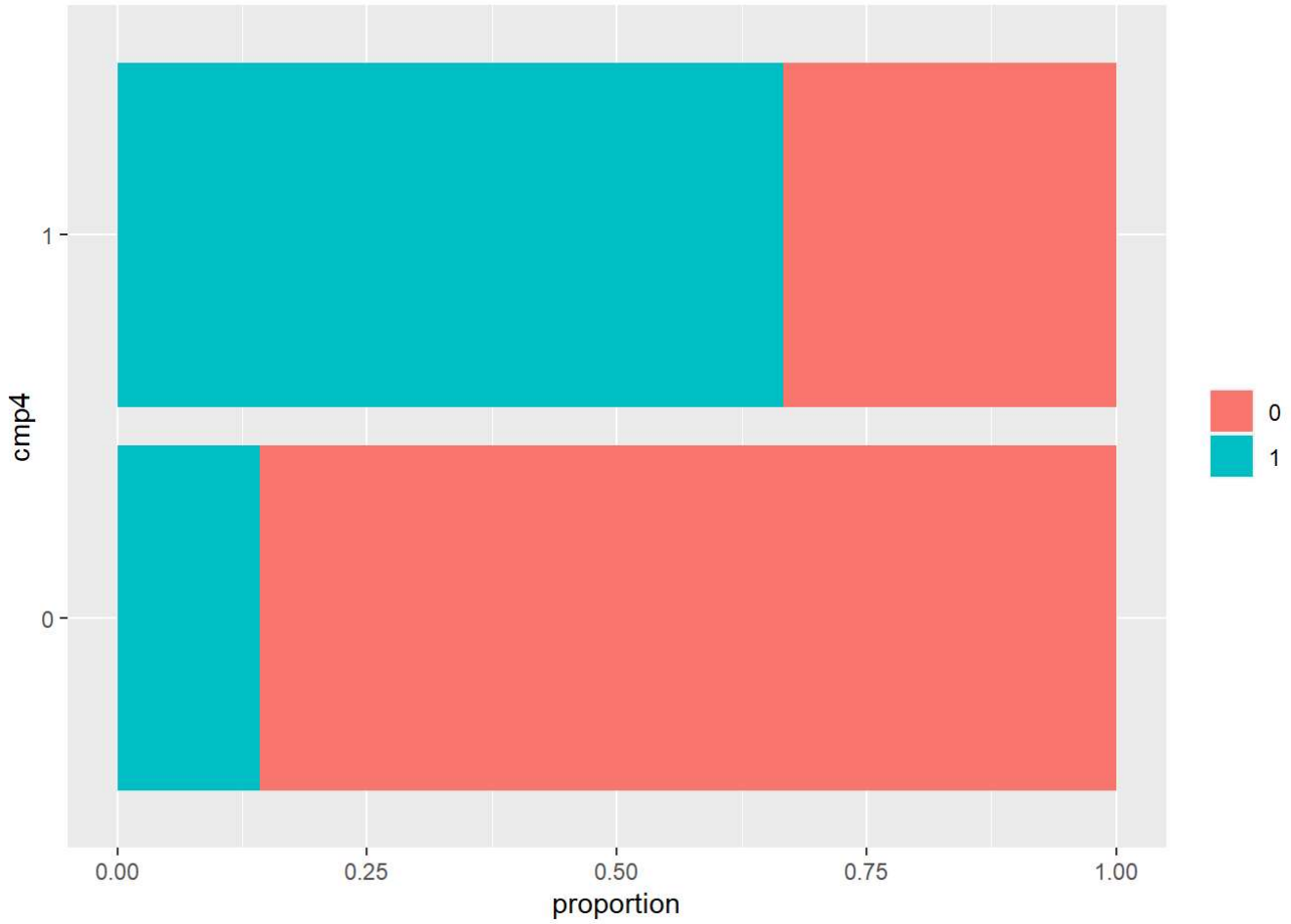

education
PhD
Master
Graduation
Basic
2nCycle
0.00
0.25
0.50
0.75
1.00
proportion
0
1

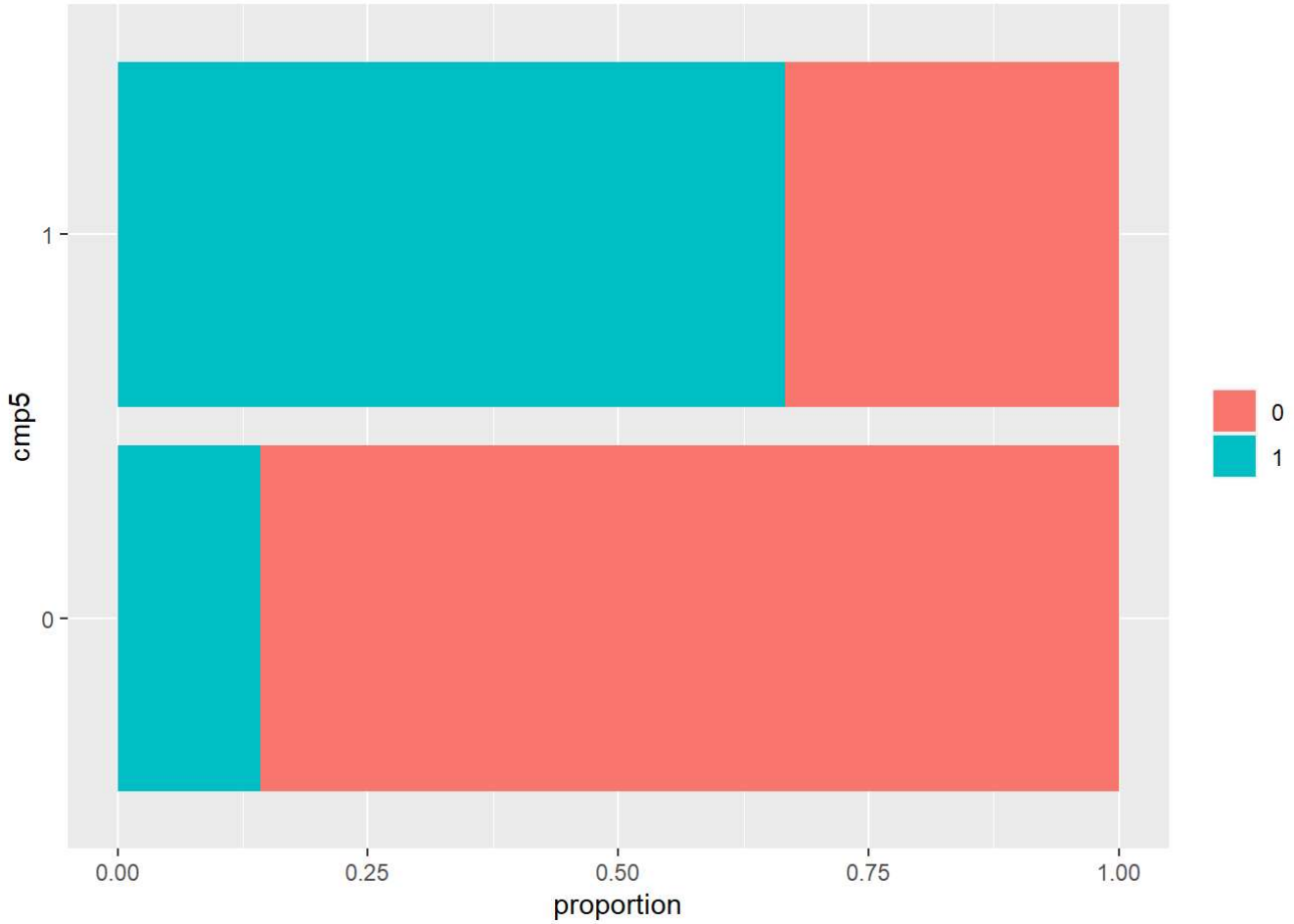

mar_stat
YOLO
Widow
Single
Partner
Married
Divorce
Alone
Absurd
0.00
0.25
0.50
0.75
1.00
proportion
0
1

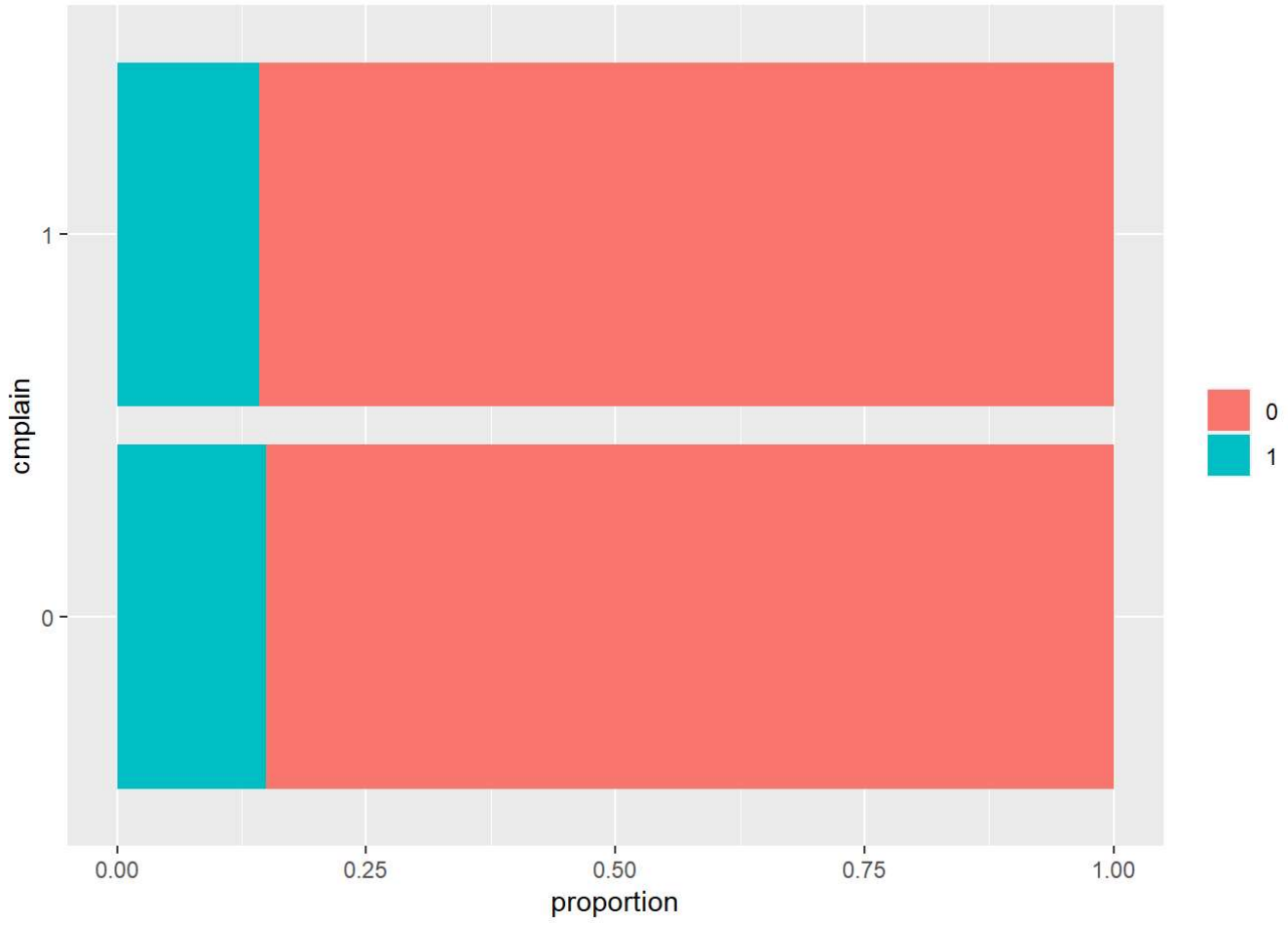

cmp1
1
0
0.00
0.25
0.50
0.75
1.00
proportion
0
1


cmp2
1
0
0.00
0.25
0.50
0.75
1.00
proportion
0
1


cmp3
1
0
0.00
0.25
0.50
0.75
1.00
proportion
0
1


cmp4
1
0
0.00
0.25
0.50
0.75
1.00
proportion
0
1





create clusters

```
clusters <- retail

# create dummy variables for gender and promotional class

clusters$phd <- ifelse(clusters$education == 'PhD', 1, 0)
clusters$master <- ifelse(clusters$education == 'Master', 1, 0)
clusters$graduation <- ifelse(clusters$education == 'Graduation', 1, 0)
clusters$basic <- ifelse(clusters$education == 'Basic', 1, 0)
clusters$X2nCycle <- ifelse(clusters$education == '2nCycle', 1, 0)

clusters$yolo <- ifelse(clusters$mar_stat == 'YOLO', 1, 0)
clusters$widow <- ifelse(clusters$mar_stat == 'Widow', 1, 0)
clusters$single <- ifelse(clusters$mar_stat == 'Single', 1, 0)
clusters$partner <- ifelse(clusters$mar_stat == 'Partner', 1, 0)
clusters$married <- ifelse(clusters$mar_stat == 'Married', 1, 0)
clusters$divorce <- ifelse(clusters$mar_stat == 'Divorce', 1, 0)
clusters$alone <- ifelse(clusters$mar_stat == 'Alone', 1, 0)
clusters$absurd <- ifelse(clusters$mar_stat == 'Absurd', 1, 0)

clusters$cmp1 <- as.numeric(clusters$cmp1)
clusters$cmp2 <- as.numeric(clusters$cmp2)
clusters$cmp3 <- as.numeric(clusters$cmp3)
clusters$cmp4 <- as.numeric(clusters$cmp4)
clusters$cmp5 <- as.numeric(clusters$cmp5)
clusters$cmplain <- as.numeric(clusters$cmplain)

#standardize numeric variables

clusters$birth <- scale(clusters$birth)
clusters$income <- scale(clusters$income)
clusters$kids <- scale(clusters$kids)
clusters$teens <- scale(clusters$teens)
clusters$recency <- scale(clusters$recency)
clusters$wines <- scale(clusters$wines)
clusters$fruits <- scale(clusters$fruits)
clusters$meat <- scale(clusters$meat)
clusters$fish <- scale(clusters$fish)
clusters$sweets <- scale(clusters$sweets)
clusters$gold <- scale(clusters$gold)
clusters$deals <- scale(clusters$deals)
clusters$web <- scale(clusters$web)
clusters$catalog <- scale(clusters$catalog)
clusters$store <- scale(clusters$store)
clusters$visits <- scale(clusters$visits)

clusters %>% skim()
```

Data summary

Name	Piped data
Number of rows	2240
Number of columns	42
_____	
Column type frequency:	
character	3
factor	1
numeric	38
_____	
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
education	0	1	3	10	0	5	0
mar_stat	0	1	4	7	0	8	0
dt_customer	0	1	8	10	0	663	0

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
response	0	1	FALSE	2	0: 1904, 1: 336

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
id	0	1	5592.16	3246.66	0.00	2828.25	5458.50	8427.75	11191.00	
birth	0	1	0.00	1.00	-6.33	-0.82	0.10	0.68	2.27	
income	0	1	0.00	1.00	-2.02	-0.67	-0.03	0.64	24.54	
kids	0	1	0.00	1.00	-0.83	-0.83	-0.83	1.03	2.89	
teens	0	1	0.00	1.00	-0.93	-0.93	-0.93	0.91	2.74	
recency	0	1	0.00	1.00	-1.70	-0.87	0.00	0.86	1.72	
wines	0	1	0.00	1.00	-0.90	-0.83	-0.39	0.60	3.53	
fruits	0	1	0.00	1.00	-0.66	-0.64	-0.46	0.17	4.34	
meat	0	1	0.00	1.00	-0.74	-0.67	-0.44	0.29	6.90	
fish	0	1	0.00	1.00	-0.69	-0.63	-0.47	0.23	4.05	
sweets	0	1	0.00	1.00	-0.66	-0.63	-0.46	0.14	5.72	
gold	0	1	0.00	1.00	-0.84	-0.67	-0.38	0.23	6.10	
deals	0	1	0.00	1.00	-1.20	-0.69	-0.17	0.35	6.56	
web	0	1	0.00	1.00	-1.47	-0.75	-0.03	0.69	8.25	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
catalog	0	1	0.00	1.00	-0.91	-0.91	-0.23	0.46	8.67	
store	0	1	0.00	1.00	-1.78	-0.86	-0.24	0.68	2.22	
visits	0	1	0.00	1.00	-2.19	-0.95	0.28	0.69	6.05	
cmp3	0	1	0.01	0.11	0.00	0.00	0.00	0.00	1.00	
cmp4	0	1	0.01	0.11	0.00	0.00	0.00	0.00	1.00	
cmp5	0	1	0.01	0.11	0.00	0.00	0.00	0.00	1.00	
cmp1	0	1	0.06	0.25	0.00	0.00	0.00	0.00	1.00	
cmp2	0	1	0.01	0.11	0.00	0.00	0.00	0.00	1.00	
cmplain	0	1	0.01	0.10	0.00	0.00	0.00	0.00	1.00	
z_cost	0	1	3.00	0.00	3.00	3.00	3.00	3.00	3.00	
z_rev	0	1	11.00	0.00	11.00	11.00	11.00	11.00	11.00	
phd	0	1	0.22	0.41	0.00	0.00	0.00	0.00	1.00	
master	0	1	0.17	0.37	0.00	0.00	0.00	0.00	1.00	
graduation	0	1	0.50	0.50	0.00	0.00	1.00	1.00	1.00	
basic	0	1	0.02	0.15	0.00	0.00	0.00	0.00	1.00	
X2nCycle	0	1	0.09	0.29	0.00	0.00	0.00	0.00	1.00	
yolo	0	1	0.00	0.03	0.00	0.00	0.00	0.00	1.00	
widow	0	1	0.03	0.18	0.00	0.00	0.00	0.00	1.00	
single	0	1	0.21	0.41	0.00	0.00	0.00	0.00	1.00	
partner	0	1	0.26	0.44	0.00	0.00	0.00	1.00	1.00	
married	0	1	0.39	0.49	0.00	0.00	0.00	1.00	1.00	
divorce	0	1	0.10	0.30	0.00	0.00	0.00	0.00	1.00	
alone	0	1	0.00	0.04	0.00	0.00	0.00	0.00	1.00	
absurd	0	1	0.00	0.03	0.00	0.00	0.00	0.00	1.00	

```
# remove redundant and rejected variables
retail_clusters = subset(clusters, select= -c(id, dt_customer, z_cost, z_rev, education, mar_stat, respons
          e))


head(retail_clusters)
```

birth	income	kids	teens	recency	wines	fruits	meat	
<dbl[,1]>	<dbl[,1]>	<dbl[,1]>	<dbl[,1]>	<dbl[,1]>	<dbl[,1]>	<dbl[,1]>	<dbl[,1]>	<c
-0.9851248	0.2356432	-0.8250334	-0.9296868	0.3069707	0.9835616	1.5512306	1.6793274	2.46
-1.2354571	-0.2354016	1.0323283	0.9067316	-0.3835785	-0.8702852	-0.6361591	-0.7130662	-0.65
-0.3175719	0.7738261	-0.8250334	-0.9296868	-0.7979081	0.3626418	0.5706766	-0.1769928	1.34
1.2678662	-1.0221272	1.0323283	-0.9296868	-0.7979081	-0.8702852	-0.5607319	-0.6510412	-0.50
1.0175339	0.2418338	1.0323283	-0.9296868	1.5499594	-0.3889980	0.4198221	-0.2168660	0.15
-0.1506837	0.4103779	-0.8250334	0.9067316	-1.1431827	0.6419072	0.3946797	-0.3054732	-0.68

6 rows | 1-9 of 35 columns

```
skim(retail_clusters)
```

Data summary

Name	retail_clusters
Number of rows	2240
Number of columns	35
_____	
Column type frequency:	
numeric	35
_____	
Group variables	None

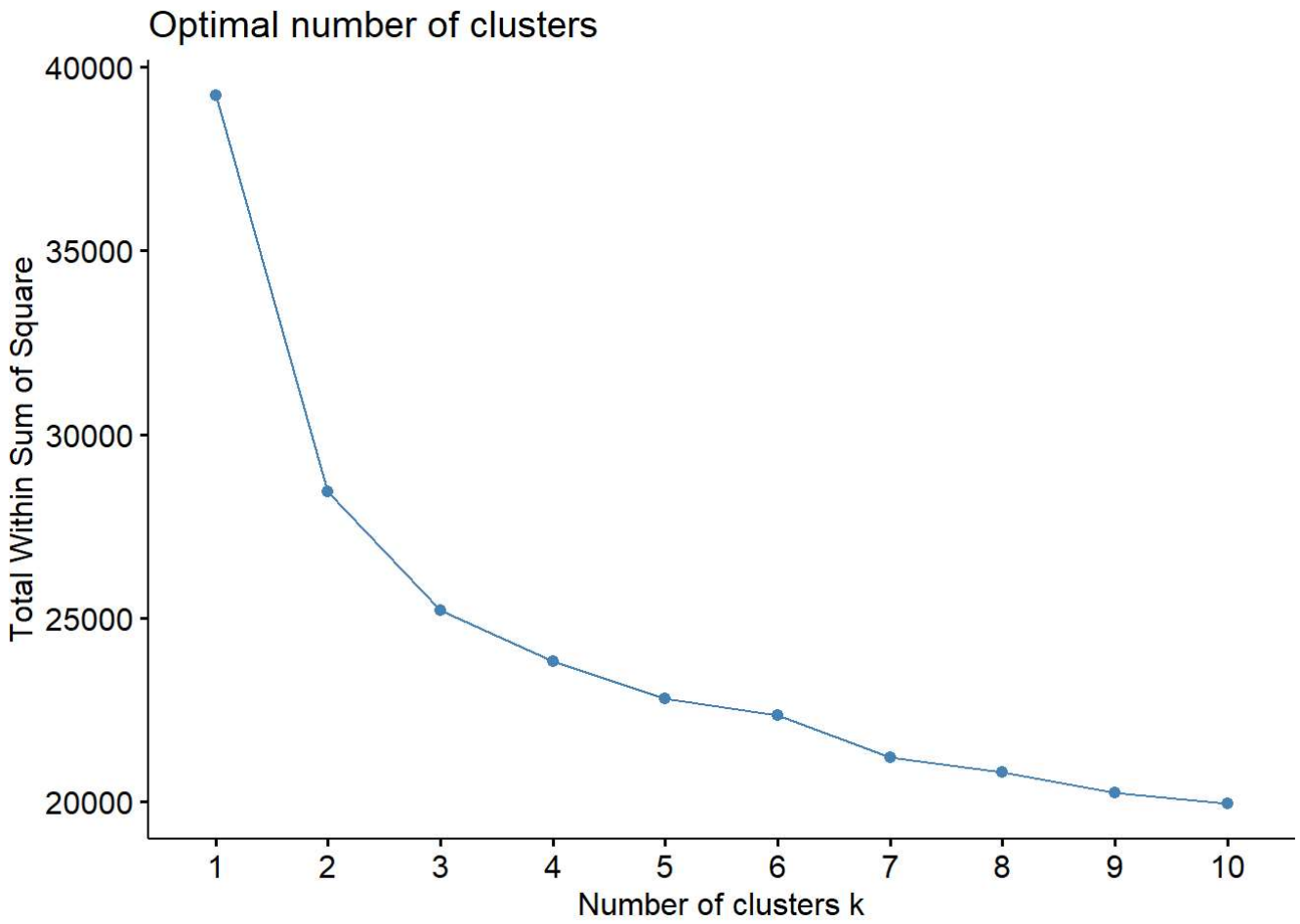
Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
birth	0	1	0.00	1.00	-6.33	-0.82	0.10	0.68	2.27	▁▂▅▇
income	0	1	0.00	1.00	-2.02	-0.67	-0.03	0.64	24.54	▇▁
kids	0	1	0.00	1.00	-0.83	-0.83	-0.83	1.03	2.89	▇▁▆▁
teens	0	1	0.00	1.00	-0.93	-0.93	-0.93	0.91	2.74	▇▁▆▁
recency	0	1	0.00	1.00	-1.70	-0.87	0.00	0.86	1.72	▇▇
wines	0	1	0.00	1.00	-0.90	-0.83	-0.39	0.60	3.53	▇▂▁
fruits	0	1	0.00	1.00	-0.66	-0.64	-0.46	0.17	4.34	▇▁
meat	0	1	0.00	1.00	-0.74	-0.67	-0.44	0.29	6.90	▇▁
fish	0	1	0.00	1.00	-0.69	-0.63	-0.47	0.23	4.05	▇▁
sweets	0	1	0.00	1.00	-0.66	-0.63	-0.46	0.14	5.72	▇▁
gold	0	1	0.00	1.00	-0.84	-0.67	-0.38	0.23	6.10	▇▁
deals	0	1	0.00	1.00	-1.20	-0.69	-0.17	0.35	6.56	▇▂
web	0	1	0.00	1.00	-1.47	-0.75	-0.03	0.69	8.25	▇▃
catalog	0	1	0.00	1.00	-0.91	-0.91	-0.23	0.46	8.67	▇▂
store	0	1	0.00	1.00	-1.78	-0.86	-0.24	0.68	2.22	▂▅▂▃▇
visits	0	1	0.00	1.00	-2.19	-0.95	0.28	0.69	6.05	▂▇▃▁
cmp3	0	1	0.01	0.11	0.00	0.00	0.00	0.00	1.00	▇▁
cmp4	0	1	0.01	0.11	0.00	0.00	0.00	0.00	1.00	▇▁
cmp5	0	1	0.01	0.11	0.00	0.00	0.00	0.00	1.00	▇▁
cmp1	0	1	0.06	0.25	0.00	0.00	0.00	0.00	1.00	▇▁
cmp2	0	1	0.01	0.11	0.00	0.00	0.00	0.00	1.00	▇▁

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
cmplain	0	1	0.01	0.10	0.00	0.00	0.00	0.00	1.00	▇▁▁▁▁
phd	0	1	0.22	0.41	0.00	0.00	0.00	0.00	1.00	▇▁▁▁▂
master	0	1	0.17	0.37	0.00	0.00	0.00	0.00	1.00	▇▁▁▁▂
graduation	0	1	0.50	0.50	0.00	0.00	1.00	1.00	1.00	▇▁▁▁▇
basic	0	1	0.02	0.15	0.00	0.00	0.00	0.00	1.00	▇▁▁▁▁
X2nCycle	0	1	0.09	0.29	0.00	0.00	0.00	0.00	1.00	▇▁▁▁▁
yolo	0	1	0.00	0.03	0.00	0.00	0.00	0.00	1.00	▇▁▁▁▁
widow	0	1	0.03	0.18	0.00	0.00	0.00	0.00	1.00	▇▁▁▁▁
single	0	1	0.21	0.41	0.00	0.00	0.00	0.00	1.00	▇▁▁▁▂
partner	0	1	0.26	0.44	0.00	0.00	0.00	0.01	1.00	▇▁▁▁▂
married	0	1	0.39	0.49	0.00	0.00	0.00	1.00	1.00	▇▁▁▁▇
divorce	0	1	0.10	0.30	0.00	0.00	0.00	0.00	1.00	▇▁▁▁▁
alone	0	1	0.00	0.04	0.00	0.00	0.00	0.00	1.00	▇▁▁▁▁
absurd	0	1	0.00	0.03	0.00	0.00	0.00	0.00	1.00	▇▁▁▁▁

visually choose number of clusters (Elbow Plot)

```
# how many clusters


fviz_nbclust(retail_clusters, kmeans, method="wss")
```



build clusters

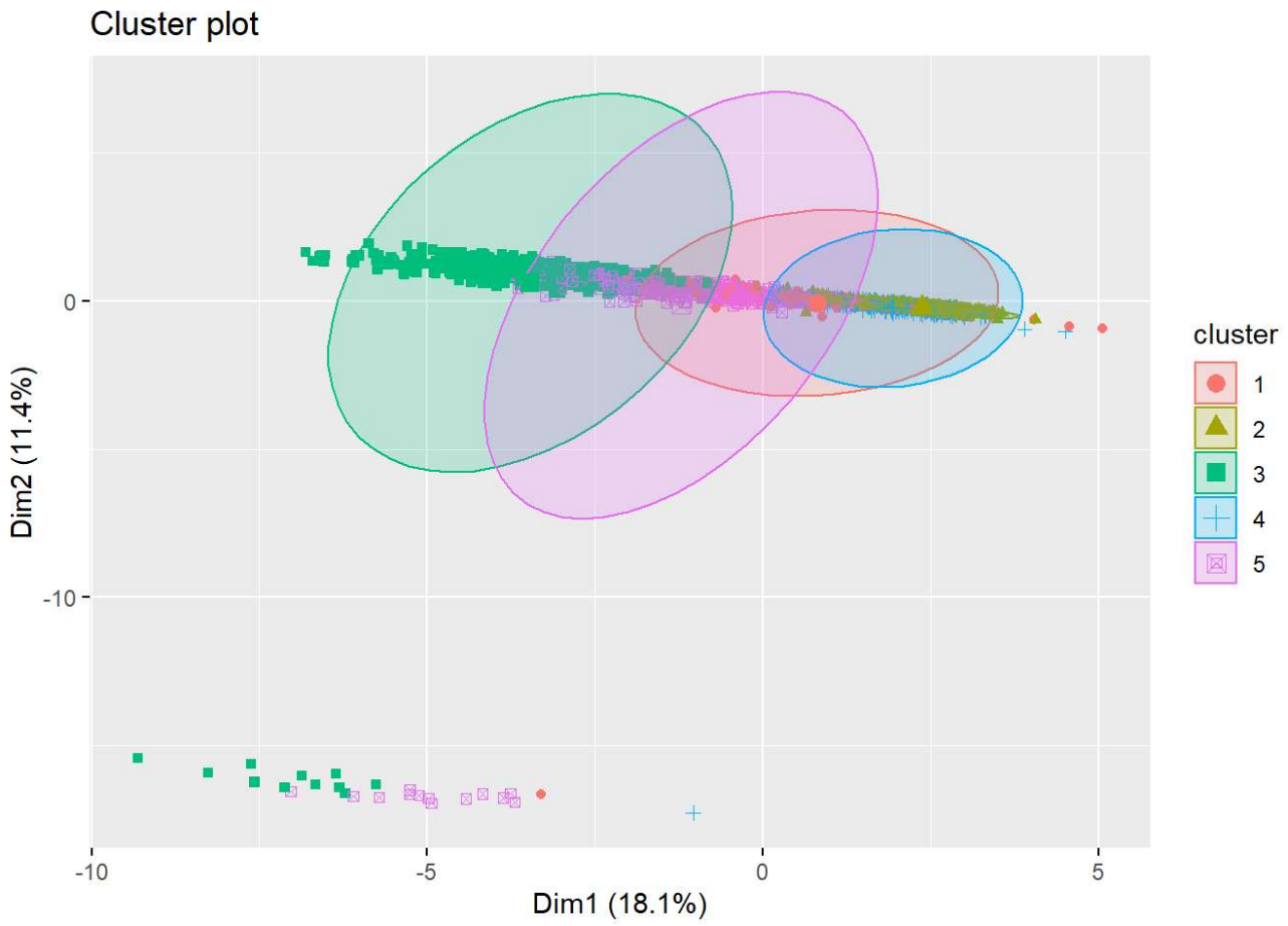
```
set.seed(1234)

clusters5 <- kmeans(retail_clusters, 5, iter.max = 200, nstart = 5)
print(clusters5)
```

```
## K-means clustering with 5 clusters of sizes 180, 537, 516, 508, 499
##
## Cluster means:
##         birth     income       kids      teens     recency       wines
## 1 -0.04081559 -0.1062944  0.7743614  0.7026851 -0.05058033  0.01186005
## 2  0.82345249 -0.9094131  0.7936728 -0.9194275 -0.03901117 -0.81634408
## 3  0.01798340  1.0127025 -0.7674408 -0.6770013  0.03784387  0.82879229
## 4 -0.48725253 -0.4197463  0.3449582  0.7910517  0.06208401 -0.72902080
## 5 -0.39399265  0.3971233 -0.6910354  0.6307168 -0.04210954  0.75937416
##        fruits        meat        fish      sweets       gold      deals
## 1 -0.33961834 -0.24027310 -0.30697793 -0.28226004  0.1544240  2.3102175
## 2 -0.50698237 -0.63125718 -0.52503033 -0.50784876 -0.5188760 -0.2848128
## 3  1.17131709  1.38448529  1.24819707  1.13350629  0.6791902 -0.5503320
## 4 -0.56177123 -0.64129961 -0.57928091 -0.55453997 -0.5823961 -0.1264293
## 5  0.02877955 -0.01278545 -0.02524583  0.04075882  0.3932568  0.1709472
##          web    catalog       store      visits        cmp3        cmp4
## 1  0.6632407 -0.1371634  0.06966811  0.81737631 0.005555556 0.005555556
## 2 -0.6919784 -0.7463331 -0.83649649  0.65768123 0.000000000 0.000000000
## 3  0.3251686  1.1492127  0.80910425 -1.09996766 0.023255814 0.023255814
## 4 -0.7212375 -0.6951975 -0.70991228  0.18187916 0.003937008 0.003937008
## 5  0.9034283  0.3720178  0.76111422 -0.05032837 0.030060120 0.030060120
##          cmp5        cmp1        cmp2     cmplain       phd    master
## 1 0.005555556 0.033333333 0.005555556 0.005555556 0.2111111 0.2000000
## 2 0.000000000 0.001862197 0.000000000 0.011173184 0.1415270 0.1471136
## 3 0.023255814 0.211240310 0.023255814 0.005813953 0.1782946 0.1414729
## 4 0.003937008 0.000000000 0.003937008 0.011811024 0.2342520 0.1948819
## 5 0.030060120 0.056112224 0.030060120 0.010020040 0.3226453 0.1663327
##   graduation       basic   X2nCycle       yolo       widow    single   partner
## 1  0.5166667 0.000000000 0.07222222 0.01111111 0.027777778 0.1555556 0.2666667
## 2  0.5083799 0.078212291 0.12476723 0.00000000 0.005586592 0.2700186 0.2346369
## 3  0.5755814 0.001937984 0.10271318 0.00000000 0.042635659 0.2538760 0.2616279
## 4  0.4645669 0.019685039 0.08661417 0.00000000 0.043307087 0.1850394 0.2913386
## 5  0.4569138 0.002004008 0.05210421 0.00000000 0.050100200 0.1643287 0.2464930
##     married    divorce       alone      absurd
## 1 0.4277778 0.10555556 0.005555556 0.000000000
## 2 0.4059590 0.08193669 0.001862197 0.000000000
## 3 0.3507752 0.08720930 0.000000000 0.003875969
## 4 0.3602362 0.11811024 0.001968504 0.000000000
## 5 0.4108216 0.12825651 0.000000000 0.000000000
##
## Clustering vector:
##    [1] 3 4 5 2 1 5 5 2 2 4 2 2 3 4 2 5 4 4 5 2 2 1 5 5 1 2 1 5 2 3 2 4 4 1 3 4 5
##   [38] 2 4 5 3 2 4 4 2 3 2 2 4 1 5 3 2 3 1 3 3 2 2 5 3 5 5 5 3 2 4 3 5 1 3 2 5 1
##   [75] 4 2 3 3 2 5 2 2 2 4 3 4 4 5 3 4 1 4 3 2 2 2 5 4 3 5 4 4 3 3 3 2 4 5 2 3 3
##  [112] 5 5 3 1 4 3 1 4 2 5 2 2 2 3 3 3 4 5 5 5 1 3 5 1 3 2 4 4 4 3 5 3 5 2 5 2 2
##  [149] 1 4 5 5 5 2 5 5 4 4 4 3 4 5 2 5 3 4 5 4 3 4 4 2 2 4 4 3 3 4 4 3 4 2 5 2 4
```

```
##  [186] 2 4 5 3 4 2 3 4 2 1 4 5 3 3 1 5 3 5 3 4 2 2 1 4 5 4 3 1 1 3 1 2 3 4 5 2 3
##  [223] 5 2 5 2 5 5 3 2 1 3 2 4 5 2 2 5 2 4 3 3 2 3 5 4 5 5 3 5 2 2 3 4 1 2 5 4 4
##  [260] 4 2 5 2 2 1 4 3 4 3 2 3 2 4 4 4 5 3 3 3 5 2 5 4 5 4 2 3 1 3 5 2 2 3 2 4 5
##  [297] 2 2 3 5 4 5 2 2 2 3 2 3 5 2 2 4 3 3 1 4 4 2 4 1 5 2 4 5 1 3 2 2 4 2 2 2 5
##  [334] 2 2 5 3 2 3 3 3 2 5 5 2 3 4 3 4 2 5 3 5 3 5 4 4 3 5 5 3 5 2 2 1 5 3 2 3 5
##  [371] 2 1 2 5 4 4 2 2 5 2 2 2 4 1 2 2 5 5 4 5 3 2 3 4 5 3 4 2 2 2 4 3 4 2 1 4 2
##  [408] 4 2 5 2 5 5 2 5 3 2 3 3 1 2 2 2 3 3 4 3 5 2 3 5 1 1 5 2 4 5 5 2 2 2 4 4 2
##  [445] 2 2 2 3 2 5 5 5 2 5 5 1 3 2 4 3 5 3 4 3 2 3 3 2 5 5 3 2 1 4 2 5 4 5 5 1 2
##  [482] 2 2 4 3 3 5 5 2 4 3 2 3 5 5 5 4 5 5 5 4 4 2 5 4 5 3 3 2 5 2 3 4 3 5 3 4 2
##  [519] 3 3 4 3 2 5 2 2 5 5 3 1 5 3 1 4 2 4 4 3 4 2 2 4 2 5 5 2 3 2 4 4 4 2 5 2 3
##  [556] 1 3 3 2 5 2 3 5 3 4 4 5 4 4 4 1 4 2 4 2 5 4 4 4 1 1 1 4 4 2 4 3 5 5 4 2 3
##  [593] 3 2 5 2 2 4 2 2 4 5 3 5 4 2 4 4 4 3 2 4 2 4 5 1 4 4 4 4 5 2 5 2 3 2 3 5 4
##  [630] 2 5 5 3 2 3 4 3 5 5 5 5 3 5 5 3 1 3 2 5 5 3 2 5 4 4 4 4 4 5 4 5 2 3 2 4 2
##  [667] 4 2 2 4 5 5 3 5 5 2 5 3 2 3 5 3 5 4 3 5 3 3 3 3 1 3 4 4 4 4 2 4 5 3 5 5 5
##  [704] 3 2 3 2 5 5 2 2 5 2 5 2 3 3 2 5 4 5 5 2 3 2 4 3 3 1 5 4 1 1 1 5 3 3 5 2 3
##  [741] 5 4 4 2 3 3 2 3 2 5 3 5 3 3 3 3 5 5 2 4 2 5 3 2 3 4 3 3 4 5 5 3 1 4 4 4 4
##  [778] 3 4 3 3 4 2 2 2 2 4 5 1 1 3 3 4 4 4 4 5 5 3 2 5 4 4 3 3 1 4 5 1 3 2 2 3 5
##  [815] 3 5 4 5 5 2 3 2 5 2 5 3 5 4 5 2 4 5 5 2 4 3 2 3 4 5 4 4 2 2 3 3 3 5 4 2 5
##  [852] 5 3 4 5 3 4 1 4 3 2 1 2 1 2 5 5 2 5 2 5 5 2 4 3 3 1 2 3 2 2 4 2 4 3 3 1 4
##  [889] 1 3 2 1 3 2 5 5 5 3 2 2 3 4 3 5 1 3 3 2 2 2 3 3 5 2 3 3 5 5 4 3 4 3 2 4 3
##  [926] 2 3 3 3 3 3 2 5 2 3 2 3 5 5 5 1 5 3 3 2 5 5 4 2 5 2 4 4 2 4 4 5 5 4 5 3 5
##  [963] 2 4 1 5 3 2 2 1 3 4 2 5 3 3 3 1 4 5 4 2 2 5 3 5 3 3 3 4 3 2 1 3 4 2 3 4 5
## [1000] 4 5 3 1 2 1 5 5 3 4 4 3 4 1 4 4 5 3 4 2 2 2 2 1 4 2 5 2 2 4 5 3 3 3 2 3 2
## [1037] 4 4 2 5 5 4 1 3 2 4 4 3 1 5 3 4 3 4 2 3 4 4 3 3 1 5 3 4 5 4 3 5 4 3 2 3 5
## [1074] 4 5 3 3 4 2 2 3 1 3 2 3 5 4 3 2 3 3 2 3 4 4 5 5 3 2 5 3 5 1 2 2 3 4 2 5 1
## [1111] 3 3 4 3 2 5 2 2 2 1 5 2 1 2 4 1 5 4 2 3 5 4 2 3 3 2 4 3 2 4 5 2 2 4 3 2 2
## [1148] 1 5 2 5 5 4 3 4 4 1 3 3 5 4 1 5 3 4 5 4 4 3 3 2 2 3 1 2 4 4 5 4 3 5 4 5 4
## [1185] 2 2 4 5 4 4 3 5 4 4 4 5 4 5 3 3 4 5 4 2 3 5 3 2 4 4 4 3 5 3 1 4 5 4 3 4 2
## [1222] 2 3 2 4 5 5 2 5 4 1 2 4 2 2 5 4 3 2 4 2 4 3 3 2 1 2 4 2 5 3 5 5 3 5 1 5 3
## [1259] 4 3 4 3 3 4 4 3 1 2 4 3 5 5 2 1 2 5 2 4 3 1 3 3 4 5 5 2 5 4 3 3 2 2 2 4 4
## [1296] 4 2 5 3 4 4 3 2 1 3 5 1 5 5 5 5 3 4 3 2 4 5 4 4 2 3 5 3 1 2 5 2 2 2 3 2 1
## [1333] 3 3 5 5 3 2 4 2 4 5 1 2 2 2 4 5 5 3 3 3 2 4 3 3 2 5 3 4 1 2 1 5 5 3 5 2 2
## [1370] 4 4 4 4 1 5 1 1 2 4 4 4 4 4 4 2 3 2 4 4 3 1 2 2 4 2 5 4 2 5 5 4 5 4 4 1 4
## [1407] 4 5 5 4 5 5 4 3 2 1 2 4 4 4 4 3 5 4 2 2 2 2 4 5 2 2 3 4 1 4 4 2 4 2 4 4 2
## [1444] 3 3 4 3 5 3 5 2 3 3 2 5 3 4 4 3 5 5 5 4 2 2 5 5 3 4 5 4 2 4 5 5 4 3 4 2 5
## [1481] 3 5 2 2 3 5 3 1 5 1 1 4 3 1 2 3 2 1 5 3 1 2 1 1 5 5 1 3 3 5 3 4 3 3 2 2 5
## [1518] 2 2 2 3 3 2 2 2 1 3 4 3 4 5 1 5 4 2 2 2 3 5 5 2 5 3 4 5 2 2 2 5 4 4 3 3 3
## [1555] 5 4 1 2 4 5 2 1 2 3 2 5 5 5 1 1 5 4 3 2 1 4 2 3 2 5 4 3 3 4 3 4 4 5 1 3 4
## [1592] 3 2 4 4 1 4 2 3 2 5 3 5 4 4 4 5 1 1 5 3 2 3 2 4 3 2 2 4 5 4 2 3 1 4 4 5 5
## [1629] 2 2 5 2 4 2 5 1 5 3 4 2 4 5 1 2 3 4 5 5 3 4 2 3 4 3 4 2 4 5 1 3 1 2 5 4 2
## [1666] 2 5 4 5 5 2 3 3 3 3 2 2 2 2 2 3 4 4 2 4 4 3 5 5 3 5 3 2 2 2 5 4 5 4 3 5 4
## [1703] 2 5 2 2 5 4 3 4 3 5 4 3 2 1 5 4 1 5 2 3 3 3 2 2 1 4 5 3 4 2 4 3 5 5 5 3 5
## [1740] 2 4 1 4 3 5 3 4 5 5 3 5 1 2 5 4 4 4 2 4 3 5 2 5 5 2 5 2 3 4 4 2 4 3 3 4 1
## [1777] 2 2 2 3 2 4 3 5 1 1 1 2 1 3 4 4 3 2 4 4 5 2 5 3 3 5 4 1 4 5 5 4 3 3 4 5 3
## [1814] 3 3 2 5 3 5 4 5 4 4 5 3 2 5 3 3 2 2 4 1 2 4 4 3 4 5 1 2 3 2 3 1 5 1 2 5 2
## [1851] 3 2 3 3 3 5 4 2 3 2 5 2 5 3 3 1 4 2 1 3 2 3 4 5 4 2 5 3 5 5 3 2 4 5 2 2 4
## [1888] 3 5 2 3 3 2 1 3 4 2 3 5 5 4 2 5 2 4 3 5 2 4 2 3 3 3 5 5 1 2 2 2 4 4 3 3 3
```

```
## [1925] 5 2 5 3 3 1 4 5 2 2 3 4 5 5 4 2 3 4 2 3 2 3 3 3 4 4 4 5 3 3 5 3 2 1 3 4 5
## [1962] 5 4 4 4 3 5 3 3 3 5 2 1 5 4 5 3 4 2 2 2 2 3 4 1 4 1 2 2 5 3 4 3 3 4 3 3 5
## [1999] 1 5 2 2 2 4 4 4 3 1 4 5 3 2 3 3 3 4 2 4 4 4 2 2 1 5 5 2 4 4 5 2 5 5 2 5 2
## [2036] 5 5 5 2 3 3 1 2 2 1 2 4 3 3 3 4 1 5 4 4 2 3 5 5 1 2 2 3 4 5 5 4 5 2 2 5 3
## [2073] 2 3 3 5 2 4 4 4 2 5 5 3 3 5 3 3 2 4 1 2 5 3 5 4 3 1 5 5 4 5 2 5 2 2 4 5 4
## [2110] 3 1 5 4 4 1 5 2 5 3 2 2 4 4 5 4 3 5 3 5 4 2 3 4 2 3 5 1 2 4 4 2 4 4 4 1 5
## [2147] 2 4 4 4 4 3 2 1 2 2 4 4 5 4 5 2 1 3 2 2 3 3 3 4 5 5 3 5 5 3 3 1 5 4 2 5 4
## [2184] 2 4 5 5 3 3 2 3 4 4 3 5 4 2 5 1 4 2 5 1 3 2 1 3 4 4 4 1 3 4 3 2 4 2 3 2 2
## [2221] 5 3 2 4 5 1 1 5 3 2 2 5 2 3 2 5 1 5 5 4
##
## Within cluster sum of squares by cluster:
## [1] 2015.090 2635.776 8776.959 3414.476 6001.319
##  (between_SS / total_SS =  41.7 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
# visualize clusters


fviz_cluster(clusters5,retail_clusters,ellipse.type="norm",geom="point")
```



explore clusters

```
cluster <- as.factor(clusters5$cluster)


clusters5
```

```
## K-means clustering with 5 clusters of sizes 180, 537, 516, 508, 499
##
## Cluster means:
##         birth     income       kids      teens     recency       wines
## 1 -0.04081559 -0.1062944  0.7743614  0.7026851 -0.05058033  0.01186005
## 2  0.82345249 -0.9094131  0.7936728 -0.9194275 -0.03901117 -0.81634408
## 3  0.01798340  1.0127025 -0.7674408 -0.6770013  0.03784387  0.82879229
## 4 -0.48725253 -0.4197463  0.3449582  0.7910517  0.06208401 -0.72902080
## 5 -0.39399265  0.3971233 -0.6910354  0.6307168 -0.04210954  0.75937416
##        fruits        meat        fish      sweets       gold      deals
## 1 -0.33961834 -0.24027310 -0.30697793 -0.28226004  0.1544240  2.3102175
## 2 -0.50698237 -0.63125718 -0.52503033 -0.50784876 -0.5188760 -0.2848128
## 3  1.17131709  1.38448529  1.24819707  1.13350629  0.6791902 -0.5503320
## 4 -0.56177123 -0.64129961 -0.57928091 -0.55453997 -0.5823961 -0.1264293
## 5  0.02877955 -0.01278545 -0.02524583  0.04075882  0.3932568  0.1709472
##          web    catalog       store      visits        cmp3        cmp4
## 1  0.6632407 -0.1371634  0.06966811  0.81737631 0.005555556 0.005555556
## 2 -0.6919784 -0.7463331 -0.83649649  0.65768123 0.000000000 0.000000000
## 3  0.3251686  1.1492127  0.80910425 -1.09996766 0.023255814 0.023255814
## 4 -0.7212375 -0.6951975 -0.70991228  0.18187916 0.003937008 0.003937008
## 5  0.9034283  0.3720178  0.76111422 -0.05032837 0.030060120 0.030060120
##          cmp5        cmp1        cmp2     cmplain       phd    master
## 1 0.005555556 0.033333333 0.005555556 0.005555556 0.2111111 0.2000000
## 2 0.000000000 0.001862197 0.000000000 0.011173184 0.1415270 0.1471136
## 3 0.023255814 0.211240310 0.023255814 0.005813953 0.1782946 0.1414729
## 4 0.003937008 0.000000000 0.003937008 0.011811024 0.2342520 0.1948819
## 5 0.030060120 0.056112224 0.030060120 0.010020040 0.3226453 0.1663327
##   graduation       basic   X2nCycle       yolo       widow    single   partner
## 1  0.5166667 0.000000000 0.07222222 0.01111111 0.027777778 0.1555556 0.2666667
## 2  0.5083799 0.078212291 0.12476723 0.00000000 0.005586592 0.2700186 0.2346369
## 3  0.5755814 0.001937984 0.10271318 0.00000000 0.042635659 0.2538760 0.2616279
## 4  0.4645669 0.019685039 0.08661417 0.00000000 0.043307087 0.1850394 0.2913386
## 5  0.4569138 0.002004008 0.05210421 0.00000000 0.050100200 0.1643287 0.2464930
##     married    divorce       alone      absurd
## 1 0.4277778 0.10555556 0.005555556 0.000000000
## 2 0.4059590 0.08193669 0.001862197 0.000000000
## 3 0.3507752 0.08720930 0.000000000 0.003875969
## 4 0.3602362 0.11811024 0.001968504 0.000000000
## 5 0.4108216 0.12825651 0.000000000 0.000000000
##
## Clustering vector:
##    [1] 3 4 5 2 1 5 5 2 2 4 2 2 3 4 2 5 4 4 5 2 2 1 5 5 1 2 1 5 2 3 2 4 4 1 3 4 5
##   [38] 2 4 5 3 2 4 4 2 3 2 2 4 1 5 3 2 3 1 3 3 2 2 5 3 5 5 5 3 2 4 3 5 1 3 2 5 1
##   [75] 4 2 3 3 2 5 2 2 2 4 3 4 4 5 3 4 1 4 3 2 2 2 5 4 3 5 4 4 3 3 3 2 4 5 2 3 3
##  [112] 5 5 3 1 4 3 1 4 2 5 2 2 2 3 3 3 4 5 5 5 1 3 5 1 3 2 4 4 4 3 5 3 5 2 5 2 2
##  [149] 1 4 5 5 5 2 5 5 4 4 4 3 4 5 2 5 3 4 5 4 3 4 4 2 2 4 4 3 3 4 4 3 4 2 5 2 4
```

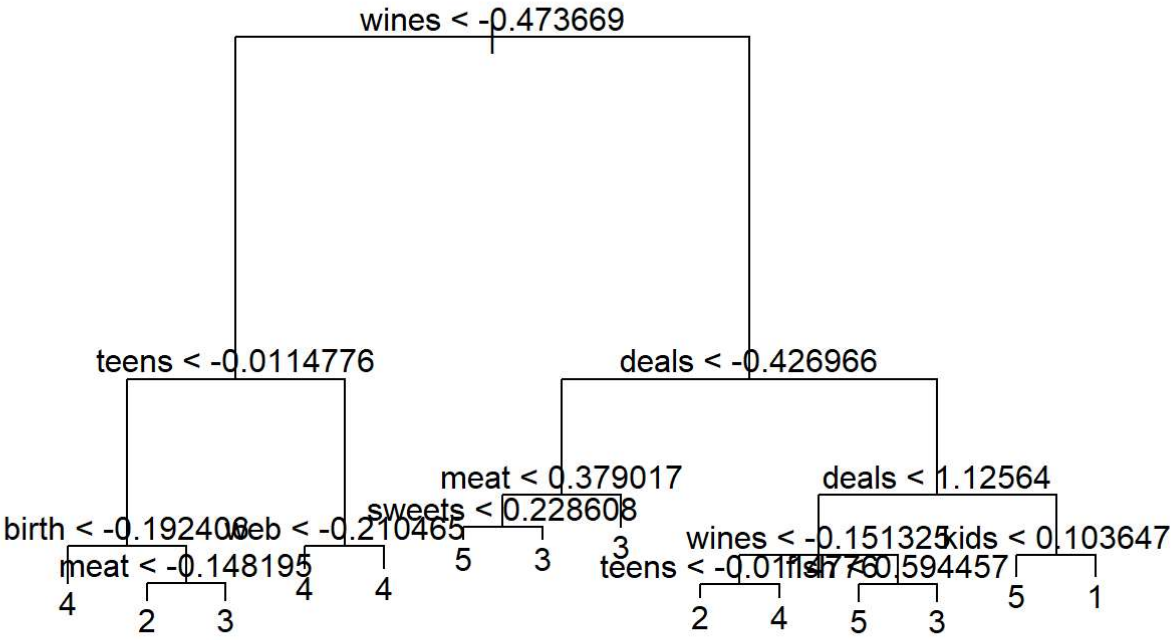
```
##  [186] 2 4 5 3 4 2 3 4 2 1 4 5 3 3 1 5 3 5 3 4 2 2 1 4 5 4 3 1 1 3 1 2 3 4 5 2 3
##  [223] 5 2 5 2 5 5 3 2 1 3 2 4 5 2 2 5 2 4 3 3 2 3 5 4 5 5 3 5 2 2 3 4 1 2 5 4 4
##  [260] 4 2 5 2 2 1 4 3 4 3 2 3 2 4 4 4 5 3 3 3 5 2 5 4 5 4 2 3 1 3 5 2 2 3 2 4 5
##  [297] 2 2 3 5 4 5 2 2 2 3 2 3 5 2 2 4 3 3 1 4 4 2 4 1 5 2 4 5 1 3 2 2 4 2 2 2 5
##  [334] 2 2 5 3 2 3 3 3 2 5 5 2 3 4 3 4 2 5 3 5 3 5 4 4 3 5 5 3 5 2 2 1 5 3 2 3 5
##  [371] 2 1 2 5 4 4 2 2 5 2 2 2 4 1 2 2 5 5 4 5 3 2 3 4 5 3 4 2 2 2 4 3 4 2 1 4 2
##  [408] 4 2 5 2 5 5 2 5 3 2 3 3 1 2 2 2 3 3 4 3 5 2 3 5 1 1 5 2 4 5 5 2 2 2 4 4 2
##  [445] 2 2 2 3 2 5 5 5 2 5 5 1 3 2 4 3 5 3 4 3 2 3 3 2 5 5 3 2 1 4 2 5 4 5 5 1 2
##  [482] 2 2 4 3 3 5 5 2 4 3 2 3 5 5 5 4 5 5 5 4 4 2 5 4 5 3 3 2 5 2 3 4 3 5 3 4 2
##  [519] 3 3 4 3 2 5 2 2 5 5 3 1 5 3 1 4 2 4 4 3 4 2 2 4 2 5 5 2 3 2 4 4 4 2 5 2 3
##  [556] 1 3 3 2 5 2 3 5 3 4 4 5 4 4 4 1 4 2 4 2 5 4 4 4 1 1 1 4 4 2 4 3 5 5 4 2 3
##  [593] 3 2 5 2 2 4 2 2 4 5 3 5 4 2 4 4 4 3 2 4 2 4 5 1 4 4 4 4 5 2 5 2 3 2 3 5 4
##  [630] 2 5 5 3 2 3 4 3 5 5 5 5 3 5 5 3 1 3 2 5 5 3 2 5 4 4 4 4 4 5 4 5 2 3 2 4 2
##  [667] 4 2 2 4 5 5 3 5 5 2 5 3 2 3 5 3 5 4 3 5 3 3 3 3 1 3 4 4 4 4 2 4 5 3 5 5 5
##  [704] 3 2 3 2 5 5 2 2 5 2 5 2 3 3 2 5 4 5 5 2 3 2 4 3 3 1 5 4 1 1 1 5 3 3 5 2 3
##  [741] 5 4 4 2 3 3 2 3 2 5 3 5 3 3 3 3 5 5 2 4 2 5 3 2 3 4 3 3 4 5 5 3 1 4 4 4 4
##  [778] 3 4 3 3 4 2 2 2 2 4 5 1 1 3 3 4 4 4 4 5 5 3 2 5 4 4 3 3 1 4 5 1 3 2 2 3 5
##  [815] 3 5 4 5 5 2 3 2 5 2 5 3 5 4 5 2 4 5 5 2 4 3 2 3 4 5 4 4 2 2 3 3 3 5 4 2 5
##  [852] 5 3 4 5 3 4 1 4 3 2 1 2 1 2 5 5 2 5 2 5 5 2 4 3 3 1 2 3 2 2 4 2 4 3 3 1 4
##  [889] 1 3 2 1 3 2 5 5 5 3 2 2 3 4 3 5 1 3 3 2 2 2 3 3 5 2 3 3 5 5 4 3 4 3 2 4 3
##  [926] 2 3 3 3 3 3 2 5 2 3 2 3 5 5 5 1 5 3 3 2 5 5 4 2 5 2 4 4 2 4 4 5 5 4 5 3 5
##  [963] 2 4 1 5 3 2 2 1 3 4 2 5 3 3 3 1 4 5 4 2 2 5 3 5 3 3 3 4 3 2 1 3 4 2 3 4 5
## [1000] 4 5 3 1 2 1 5 5 3 4 4 3 4 1 4 4 5 3 4 2 2 2 2 1 4 2 5 2 2 4 5 3 3 3 2 3 2
## [1037] 4 4 2 5 5 4 1 3 2 4 4 3 1 5 3 4 3 4 2 3 4 4 3 3 1 5 3 4 5 4 3 5 4 3 2 3 5
## [1074] 4 5 3 3 4 2 2 3 1 3 2 3 5 4 3 2 3 3 2 3 4 4 5 5 3 2 5 3 5 1 2 2 3 4 2 5 1
## [1111] 3 3 4 3 2 5 2 2 2 1 5 2 1 2 4 1 5 4 2 3 5 4 2 3 3 2 4 3 2 4 5 2 2 4 3 2 2
## [1148] 1 5 2 5 5 4 3 4 4 1 3 3 5 4 1 5 3 4 5 4 4 3 3 2 2 3 1 2 4 4 5 4 3 5 4 5 4
## [1185] 2 2 4 5 4 4 3 5 4 4 4 5 4 5 3 3 4 5 4 2 3 5 3 2 4 4 4 3 5 3 1 4 5 4 3 4 2
## [1222] 2 3 2 4 5 5 2 5 4 1 2 4 2 2 5 4 3 2 4 2 4 3 3 2 1 2 4 2 5 3 5 5 3 5 1 5 3
## [1259] 4 3 4 3 3 4 4 3 1 2 4 3 5 5 2 1 2 5 2 4 3 1 3 3 4 5 5 2 5 4 3 3 2 2 2 4 4
## [1296] 4 2 5 3 4 4 3 2 1 3 5 1 5 5 5 5 3 4 3 2 4 5 4 4 2 3 5 3 1 2 5 2 2 2 3 2 1
## [1333] 3 3 5 5 3 2 4 2 4 5 1 2 2 2 4 5 5 3 3 3 2 4 3 3 2 5 3 4 1 2 1 5 5 3 5 2 2
## [1370] 4 4 4 4 1 5 1 1 2 4 4 4 4 4 4 2 3 2 4 4 3 1 2 2 4 2 5 4 2 5 5 4 5 4 4 1 4
## [1407] 4 5 5 4 5 5 4 3 2 1 2 4 4 4 4 3 5 4 2 2 2 2 4 5 2 2 3 4 1 4 4 2 4 2 4 4 2
## [1444] 3 3 4 3 5 3 5 2 3 3 2 5 3 4 4 3 5 5 5 4 2 2 5 5 3 4 5 4 2 4 5 5 4 3 4 2 5
## [1481] 3 5 2 2 3 5 3 1 5 1 1 4 3 1 2 3 2 1 5 3 1 2 1 1 5 5 1 3 3 5 3 4 3 3 2 2 5
## [1518] 2 2 2 3 3 2 2 2 1 3 4 3 4 5 1 5 4 2 2 2 3 5 5 2 5 3 4 5 2 2 2 5 4 4 3 3 3
## [1555] 5 4 1 2 4 5 2 1 2 3 2 5 5 5 1 1 5 4 3 2 1 4 2 3 2 5 4 3 3 4 3 4 4 5 1 3 4
## [1592] 3 2 4 4 1 4 2 3 2 5 3 5 4 4 4 5 1 1 5 3 2 3 2 4 3 2 2 4 5 4 2 3 1 4 4 5 5
## [1629] 2 2 5 2 4 2 5 1 5 3 4 2 4 5 1 2 3 4 5 5 3 4 2 3 4 3 4 2 4 5 1 3 1 2 5 4 2
## [1666] 2 5 4 5 5 2 3 3 3 3 2 2 2 2 2 3 4 4 2 4 4 3 5 5 3 5 3 2 2 2 5 4 5 4 3 5 4
## [1703] 2 5 2 2 5 4 3 4 3 5 4 3 2 1 5 4 1 5 2 3 3 3 2 2 1 4 5 3 4 2 4 3 5 5 5 3 5
## [1740] 2 4 1 4 3 5 3 4 5 5 3 5 1 2 5 4 4 4 2 4 3 5 2 5 5 2 5 2 3 4 4 2 4 3 3 4 1
## [1777] 2 2 2 3 2 4 3 5 1 1 1 2 1 3 4 4 3 2 4 4 5 2 5 3 3 5 4 1 4 5 5 4 3 3 4 5 3
## [1814] 3 3 2 5 3 5 4 5 4 4 5 3 2 5 3 3 2 2 4 1 2 4 4 3 4 5 1 2 3 2 3 1 5 1 2 5 2
## [1851] 3 2 3 3 3 5 4 2 3 2 5 2 5 3 3 1 4 2 1 3 2 3 4 5 4 2 5 3 5 5 3 2 4 5 2 2 4
## [1888] 3 5 2 3 3 2 1 3 4 2 3 5 5 4 2 5 2 4 3 5 2 4 2 3 3 3 5 5 1 2 2 2 4 4 3 3 3
```

```
## [1925] 5 2 5 3 3 1 4 5 2 2 3 4 5 5 4 2 3 4 2 3 2 3 3 3 4 4 4 5 3 3 5 3 2 1 3 4 5
## [1962] 5 4 4 4 3 5 3 3 3 5 2 1 5 4 5 3 4 2 2 2 2 3 4 1 4 1 2 2 5 3 4 3 3 4 3 3 5
## [1999] 1 5 2 2 2 4 4 4 3 1 4 5 3 2 3 3 3 4 2 4 4 4 2 2 1 5 5 2 4 4 5 2 5 5 2 5 2
## [2036] 5 5 5 2 3 3 1 2 2 1 2 4 3 3 3 4 1 5 4 4 2 3 5 5 1 2 2 3 4 5 5 4 5 2 2 5 3
## [2073] 2 3 3 5 2 4 4 4 2 5 5 3 3 5 3 3 2 4 1 2 5 3 5 4 3 1 5 5 4 5 2 5 2 2 4 5 4
## [2110] 3 1 5 4 4 1 5 2 5 3 2 2 4 4 5 4 3 5 3 5 4 2 3 4 2 3 5 1 2 4 4 2 4 4 4 1 5
## [2147] 2 4 4 4 4 3 2 1 2 2 4 4 5 4 5 2 1 3 2 2 3 3 3 4 5 5 3 5 5 3 3 1 5 4 2 5 4
## [2184] 2 4 5 5 3 3 2 3 4 4 3 5 4 2 5 1 4 2 5 1 3 2 1 3 4 4 4 1 3 4 3 2 4 2 3 2 2
## [2221] 5 3 2 4 5 1 1 5 3 2 2 5 2 3 2 5 1 5 5 4
##
## Within cluster sum of squares by cluster:
## [1] 2015.090 2635.776 8776.959 3414.476 6001.319
##  (between_SS / total_SS =  41.7 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
#determine which variables are driving the cluster creation


tree.clusters=tree(cluster~.,retail_clusters)


summary(tree.clusters)
```

```
##
## Classification tree:
## tree(formula = cluster ~ ., data = retail_clusters)
## Variables actually used in tree construction:
## [1] "wines"  "teens"  "birth"  "meat"   "web"    "deals"  "sweets" "fish"
## [9] "kids"
## Number of terminal nodes:  14
## Residual mean deviance:  0.7212 = 1605 / 2226
## Misclassification error rate: 0.1299 = 291 / 2240
```

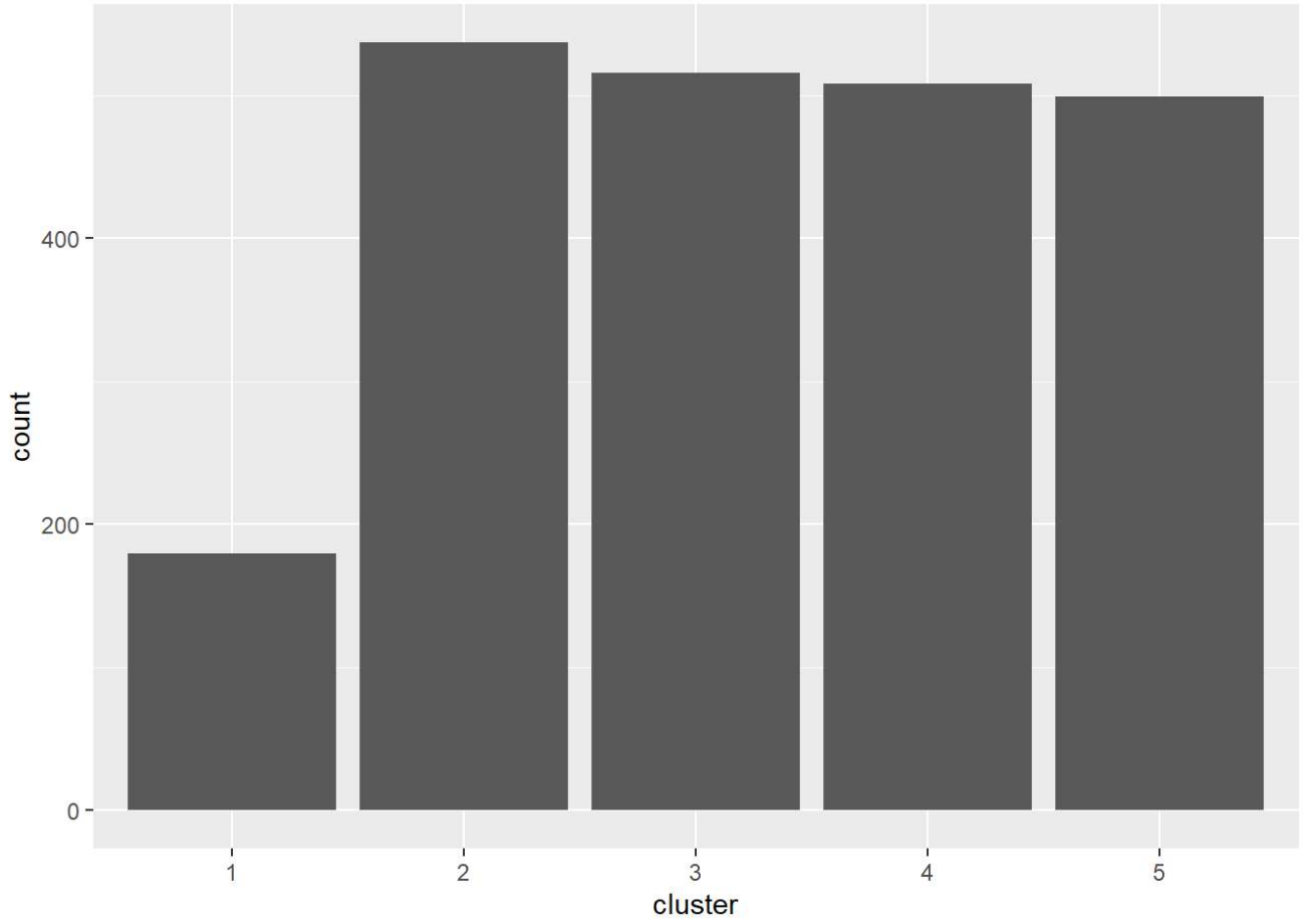
```
plot(tree.clusters)
text(tree.clusters,pretty=0)
```



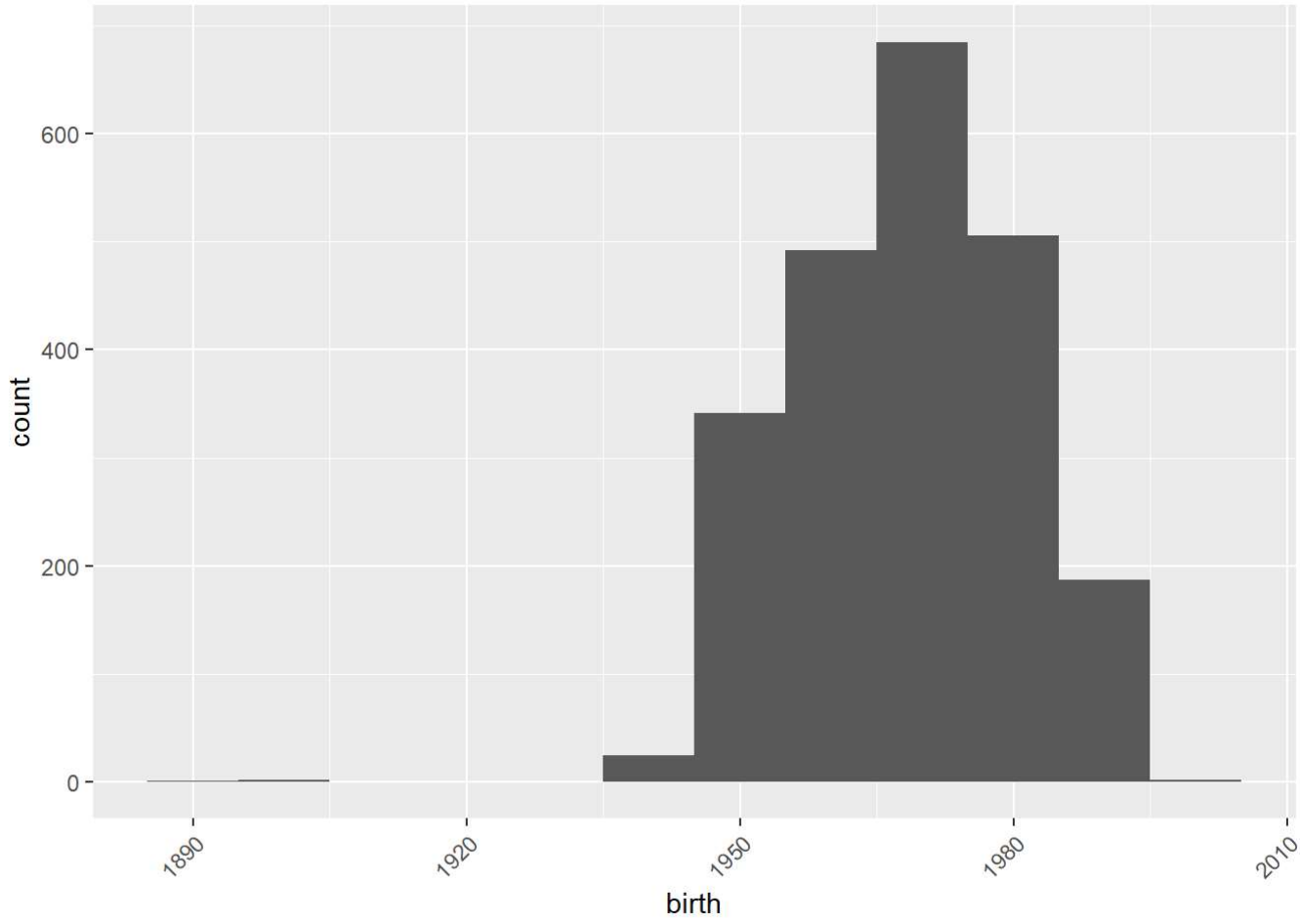

```
tree.clusters
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##  1) root 2240 6963.00 2 ( 0.080357 0.239732 0.230357 0.226786 0.222768 )
##    2) wines < -0.473669 1041 2006.00 2 ( 0.024015 0.493756 0.015370 0.439001 0.027858 )
##      4) teens < -0.0114776 590  650.50 2 ( 0.015254 0.866102 0.018644 0.072881 0.027119 )
##        8) birth < -0.192406 65  126.80 4 ( 0.000000 0.230769 0.015385 0.615385 0.138462 ) *
##        9) birth > -0.192406 525  300.20 2 ( 0.017143 0.944762 0.019048 0.005714 0.013333 )
##         18) meat < -0.148195 508  156.10 2 ( 0.007874 0.974409 0.003937 0.005906 0.007874 ) *
##         19) meat > -0.148195 17   40.37 3 ( 0.294118 0.058824 0.470588 0.000000 0.176471 ) *
##      5) teens > -0.0114776 451  345.00 4 ( 0.035477 0.006652 0.011086 0.917960 0.028825 )
##       10) web < -0.210465 403   88.50 4 ( 0.004963 0.007444 0.002481 0.982630 0.002481 ) *
##       11) web > -0.210465 48  123.00 4 ( 0.291667 0.000000 0.083333 0.375000 0.250000 ) *
##    3) wines > -0.473669 1199 2893.00 3 ( 0.129274 0.019183 0.417014 0.042535 0.391993 )
##      6) deals < -0.426966 540  644.30 3 ( 0.000000 0.005556 0.796296 0.020370 0.177778 )
##       12) meat < 0.379017 144  283.50 5 ( 0.000000 0.020833 0.381944 0.076389 0.520833 )
##         24) sweets < 0.228608 86  149.00 5 ( 0.000000 0.034884 0.116279 0.127907 0.720930 ) *
##         25) sweets > 0.228608 58   61.72 3 ( 0.000000 0.000000 0.775862 0.000000 0.224138 ) *
##       13) meat > 0.379017 396  164.20 3 ( 0.000000 0.000000 0.946970 0.000000 0.053030 ) *
##      7) deals > -0.426966 659 1550.00 5 ( 0.235205 0.030349 0.106222 0.060698 0.567527 )
##       14) deals < 1.12564 462  889.10 5 ( 0.030303 0.041126 0.136364 0.084416 0.707792 )
##         28) wines < -0.151325 108  318.50 4 ( 0.074074 0.166667 0.120370 0.333333 0.305556 )
##           56) teens < -0.0114776 34   79.78 2 ( 0.088235 0.529412 0.235294 0.000000 0.147059 ) *
##           57) teens > -0.0114776 74  160.20 4 ( 0.067568 0.000000 0.067568 0.486486 0.378378 ) *
##         29) wines > -0.151325 354  394.20 5 ( 0.016949 0.002825 0.141243 0.008475 0.830508 )
##           58) fish < 0.594457 268  160.00 5 ( 0.018657 0.003731 0.026119 0.011194 0.940299 ) *
##           59) fish > 0.594457 86  128.70 3 ( 0.011628 0.000000 0.500000 0.000000 0.488372 ) *
##       15) deals > 1.12564 197  296.90 1 ( 0.715736 0.005076 0.035533 0.005076 0.238579 )
##         30) kids < 0.103647 82  155.60 5 ( 0.329268 0.000000 0.085366 0.012195 0.573171 ) *
##         31) kids > 0.103647 115   11.48 1 ( 0.991304 0.008696 0.000000 0.000000 0.000000 ) *
```

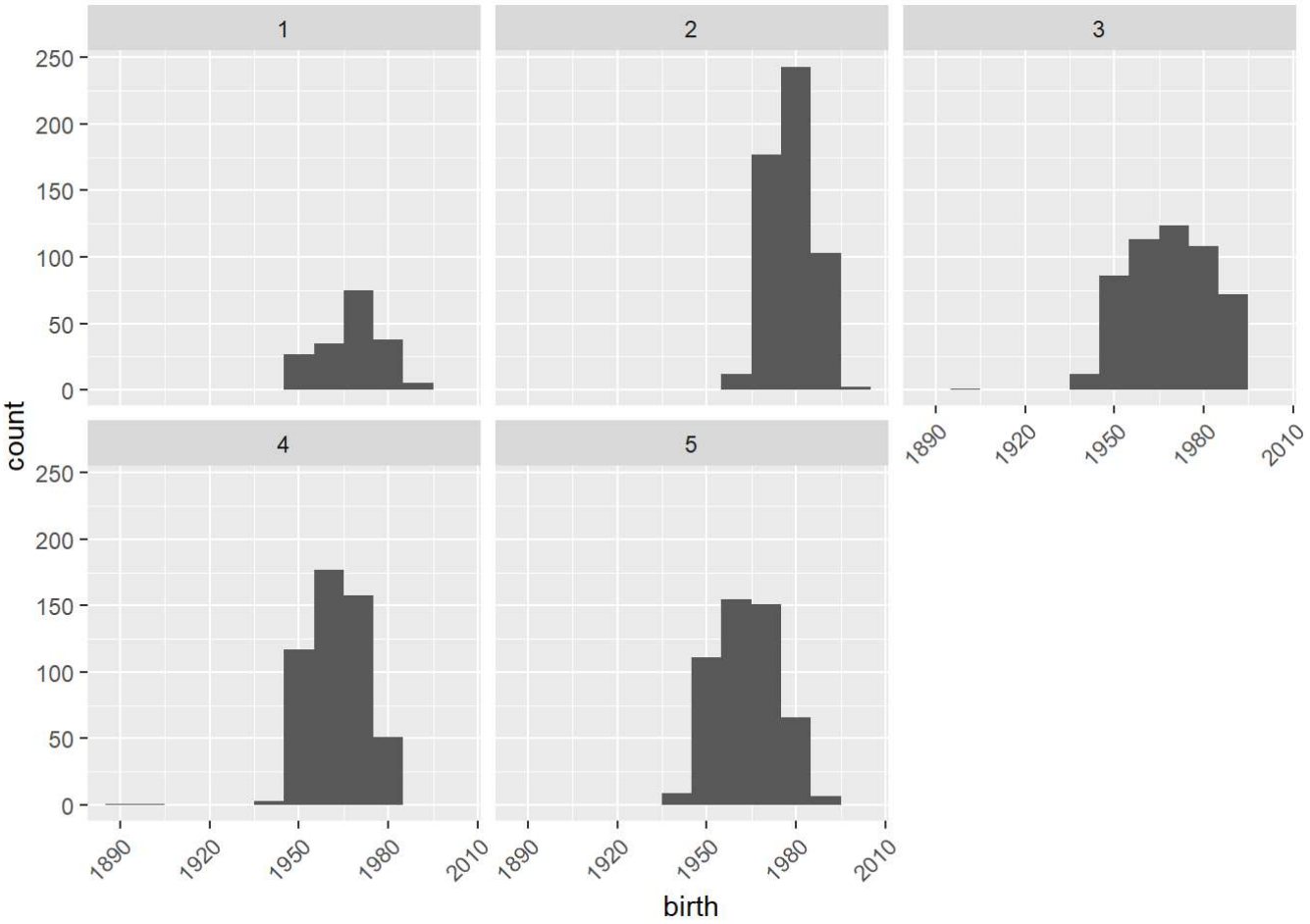
```
ggplot(retail,aes(cluster))+geom_bar()
```



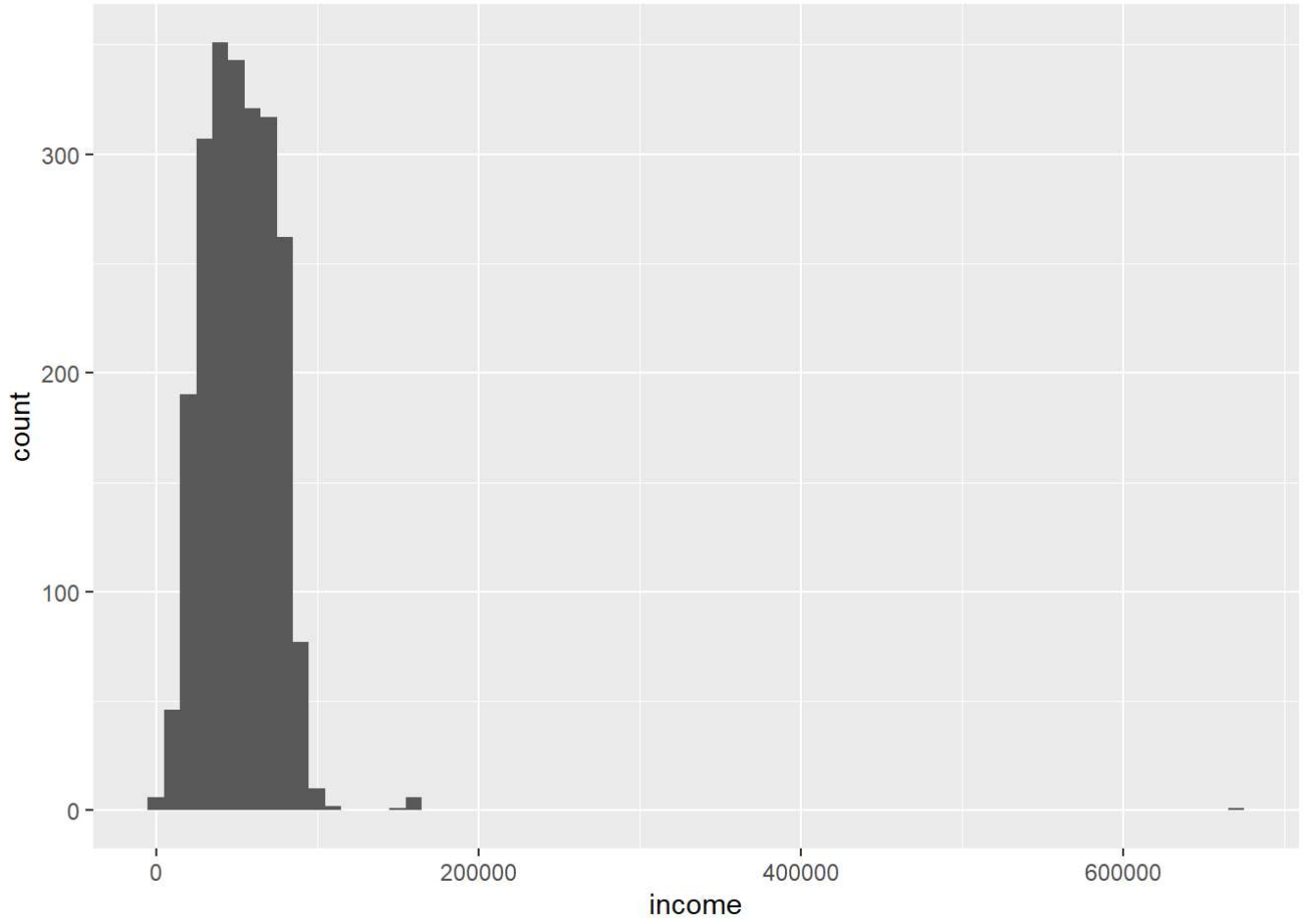

```
ggplot(retail,aes(x=birth))+geom_histogram(binwidth=10)+theme(axis.text.x=element_text(angle=45, hjust=1))
```



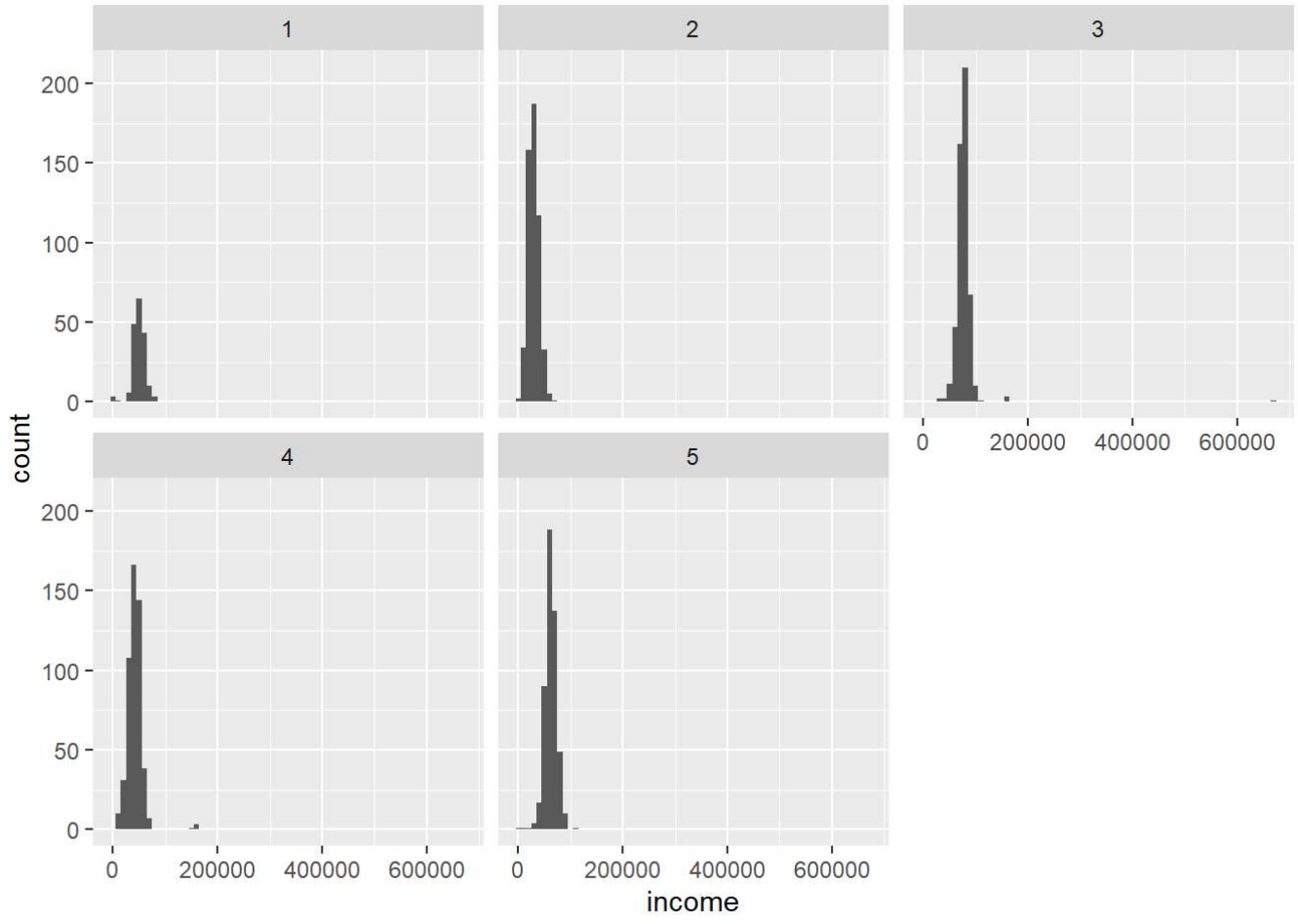
```
ggplot(retail,aes(x=birth))+geom_histogram(binwidth=10)+facet_wrap(~clusters5$cluster)+theme(axis.text.x=ele
          ment_text(angle=45, hjust=1))
```



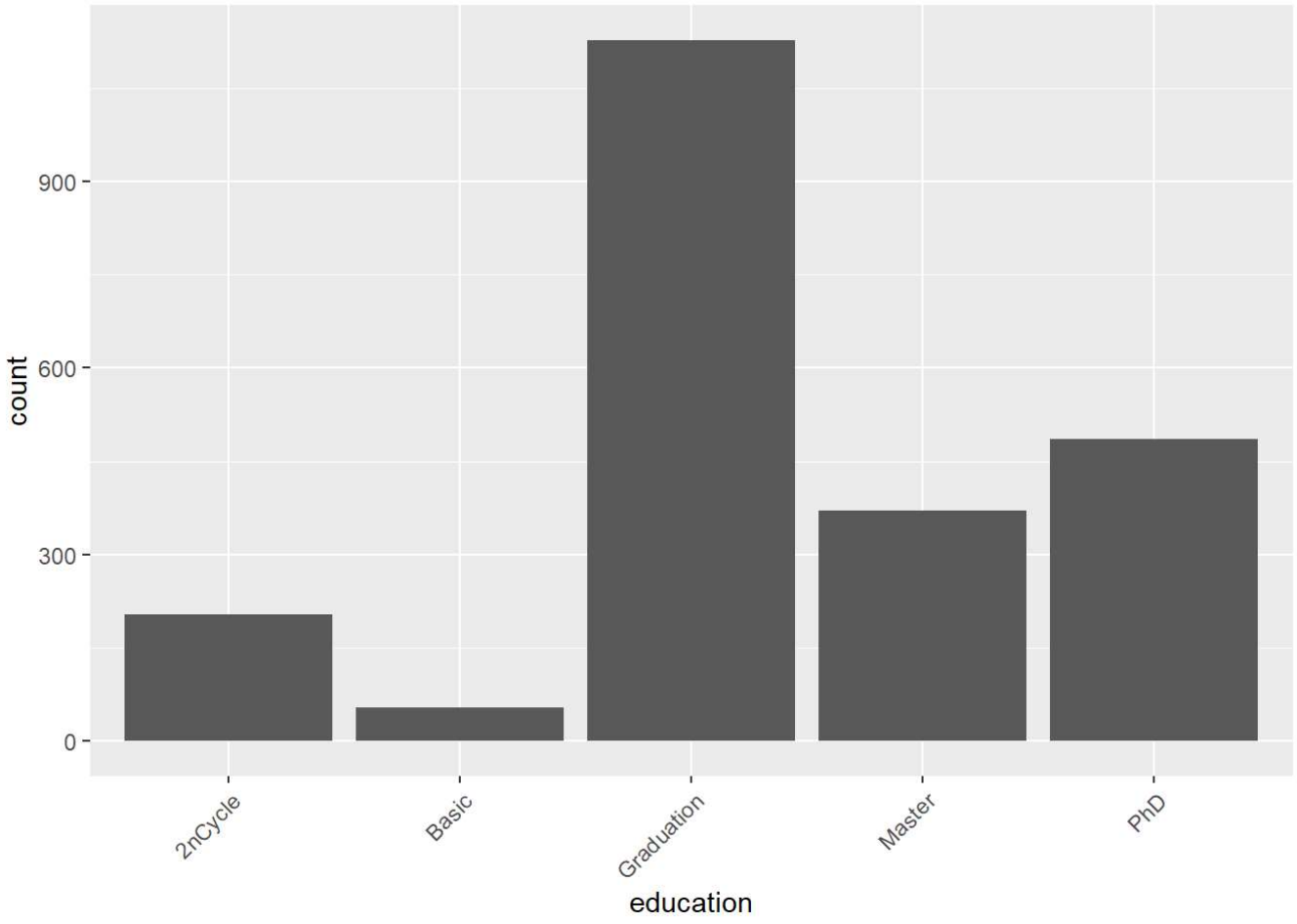

```
ggplot(retail,aes(x=income))+geom_histogram(binwidth=10000)
```



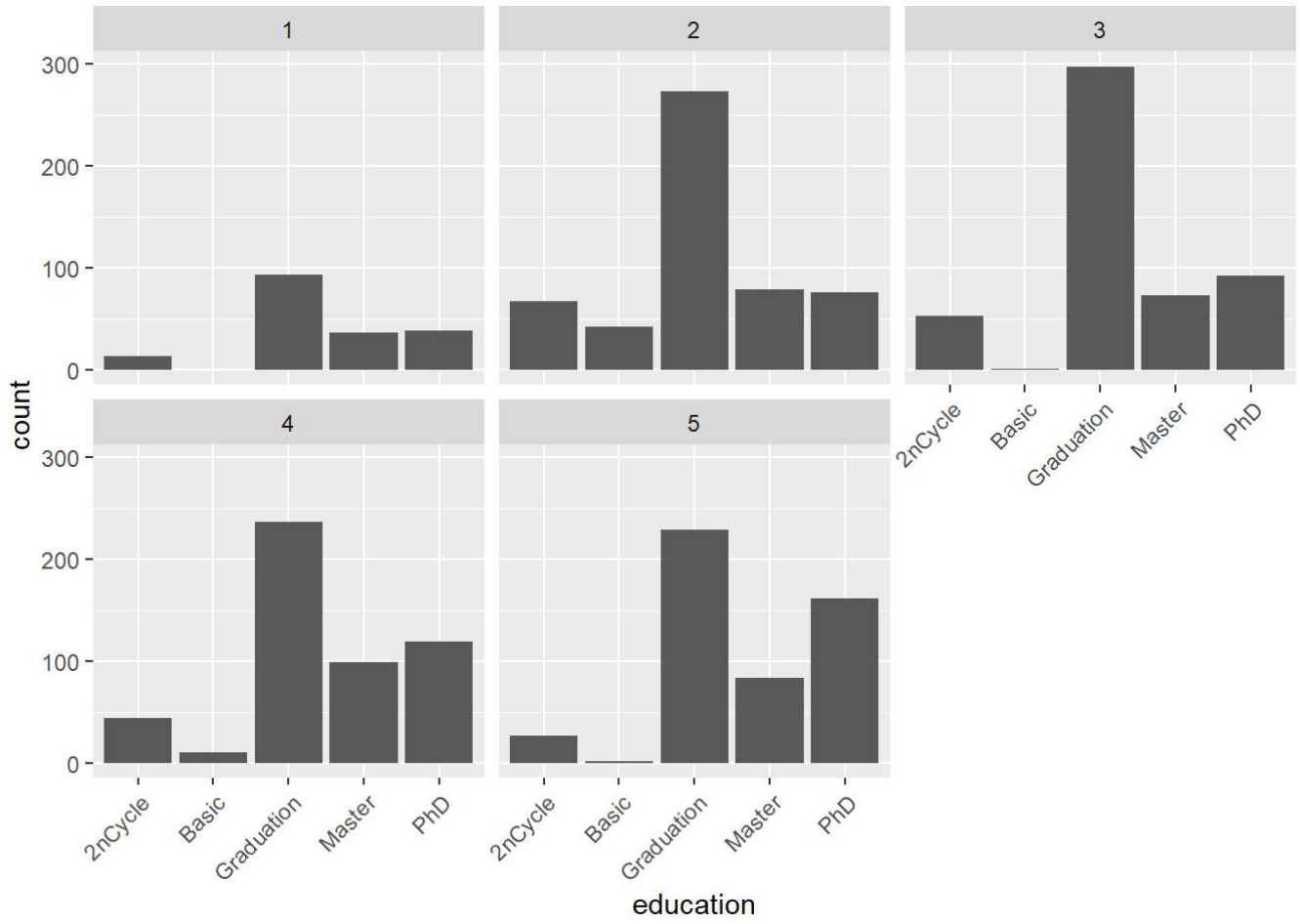

```
ggplot(retail,aes(x=income))+geom_histogram(binwidth=10000) + facet_wrap(~clusters5$cluster)
```



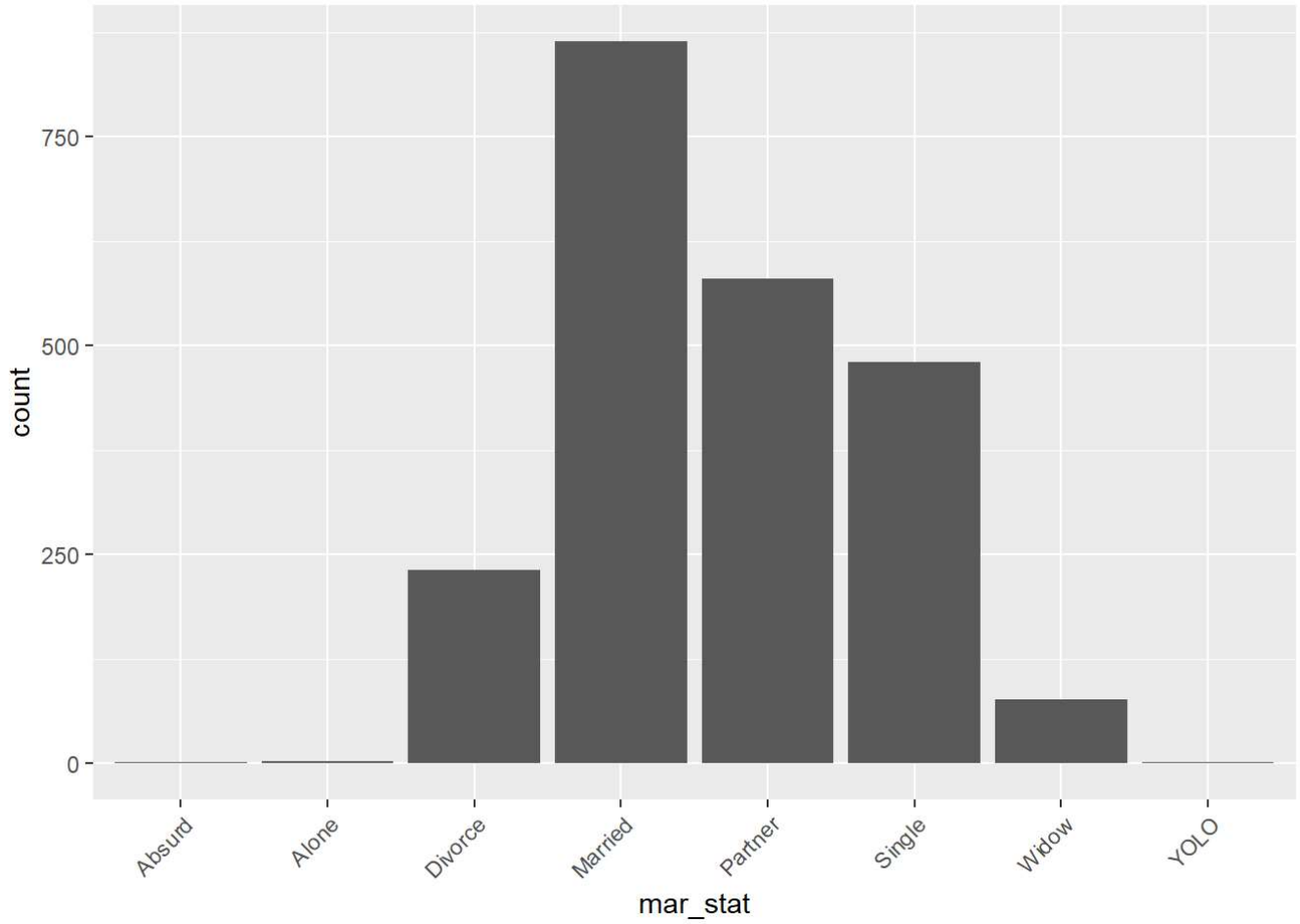
```
ggplot(retail,aes(education))+geom_bar()+theme(axis.text.x=element_text(angle=45, hjust=1))
```



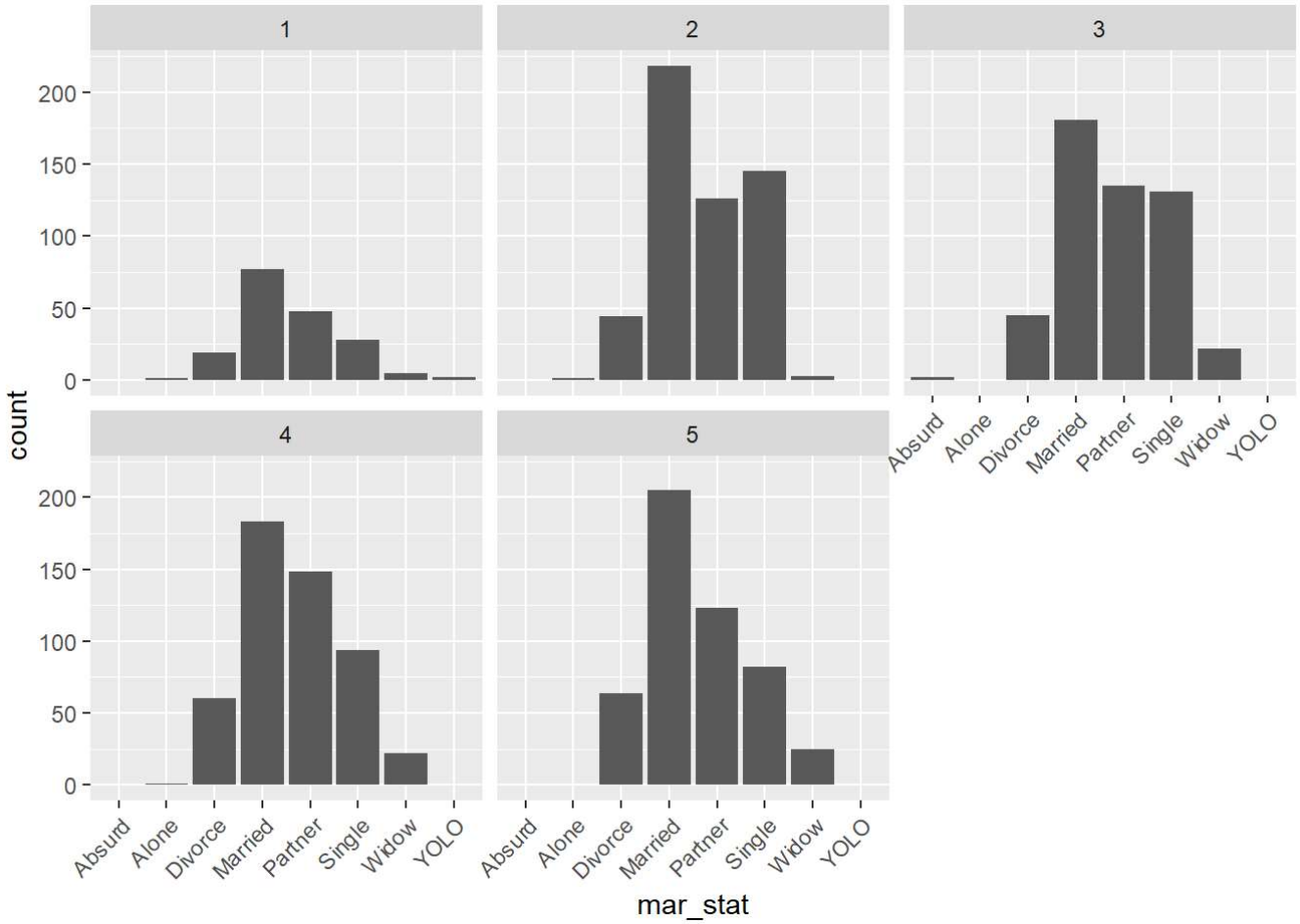

```
ggplot(retail,aes(education))+geom_bar()+facet_wrap(~clusters5$cluster)+theme(axis.text.x=element_text(angle
          =45, hjust=1))
```



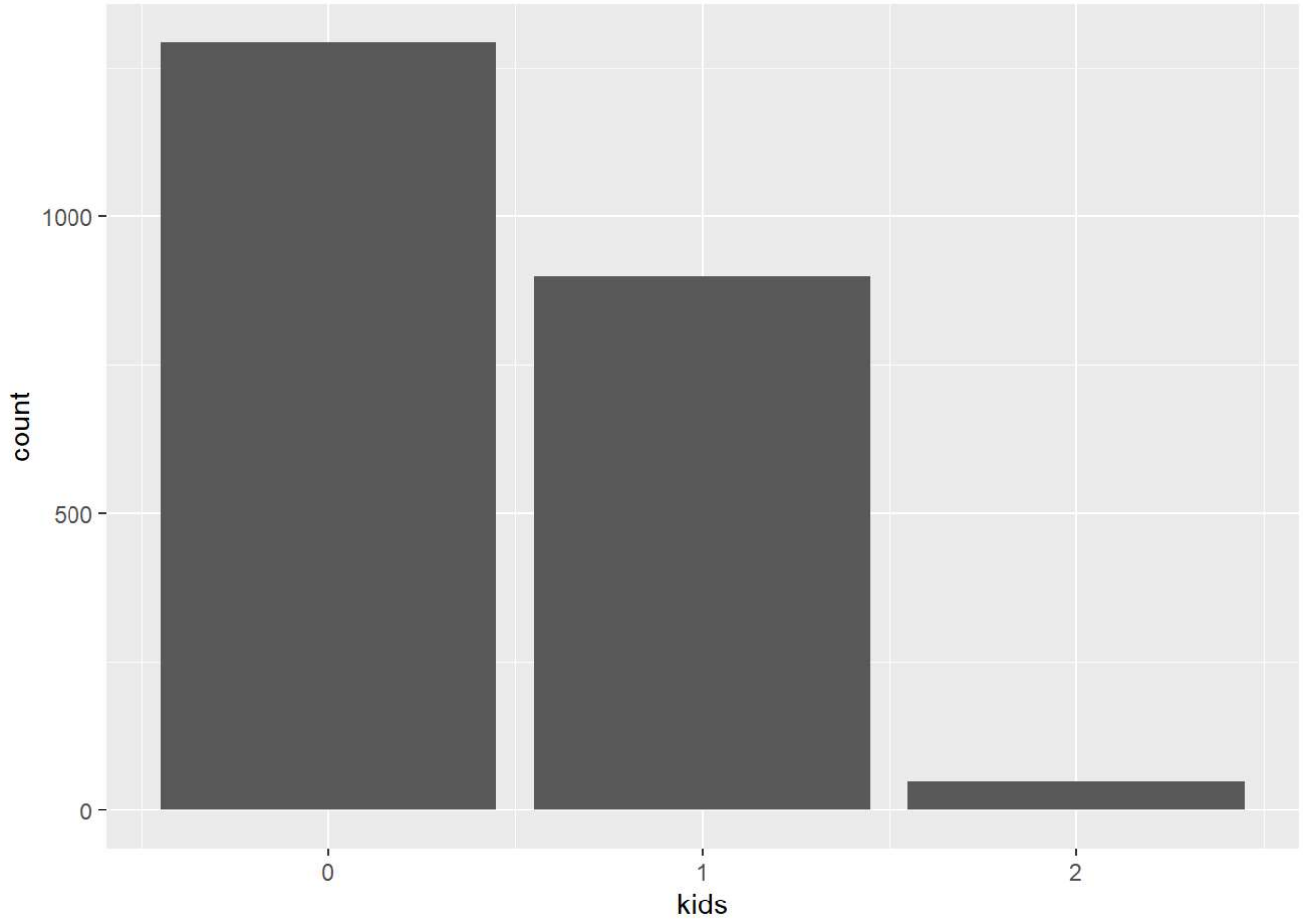

```
ggplot(retail,aes(mar_stat))+geom_bar()+theme(axis.text.x=element_text(angle=45, hjust=1))
```



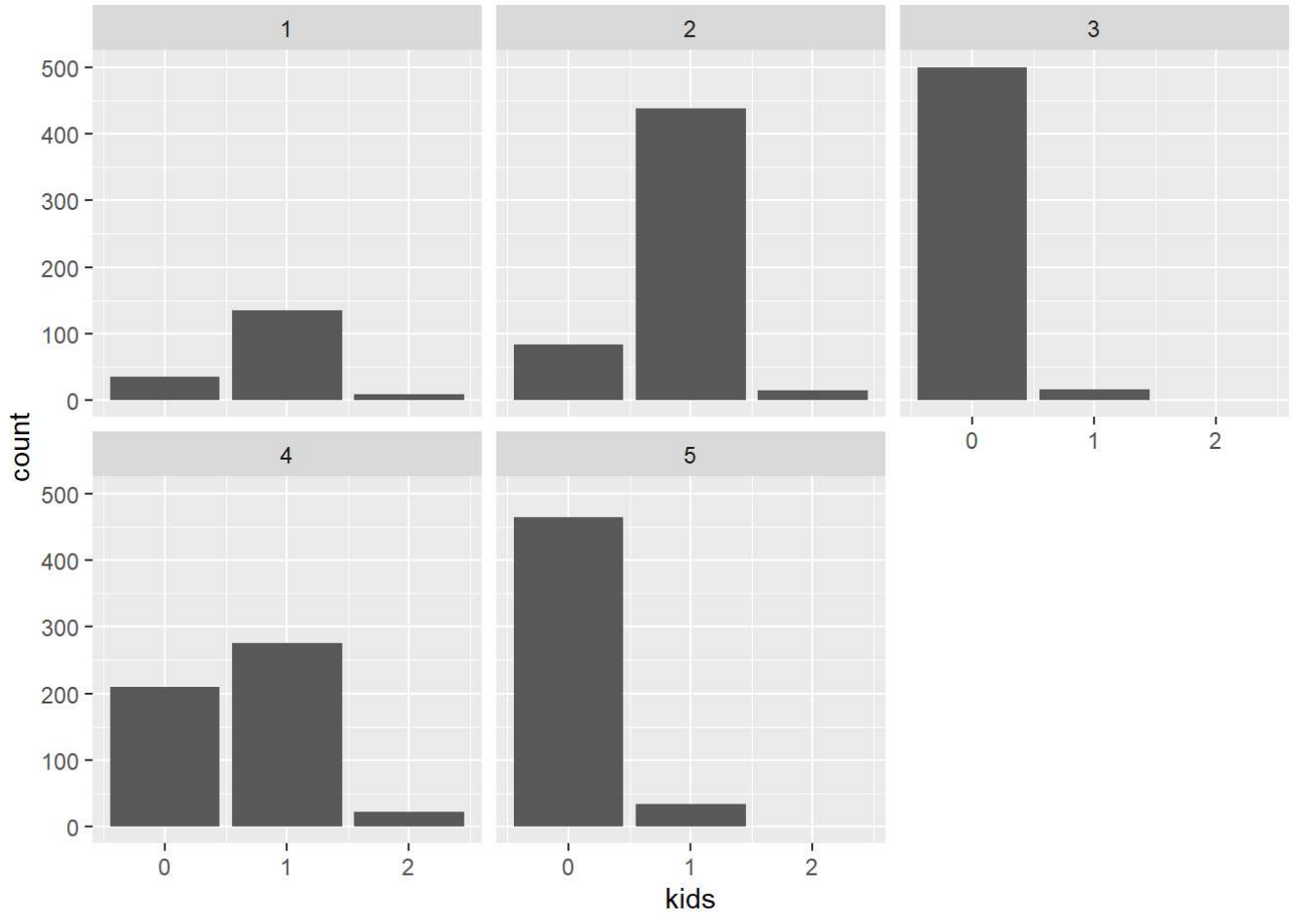
```
ggplot(retail,aes(mar_stat))+geom_bar()+facet_wrap(~clusters5$cluster)+theme(axis.text.x=element_text(angle=
          45, hjust=1))
```



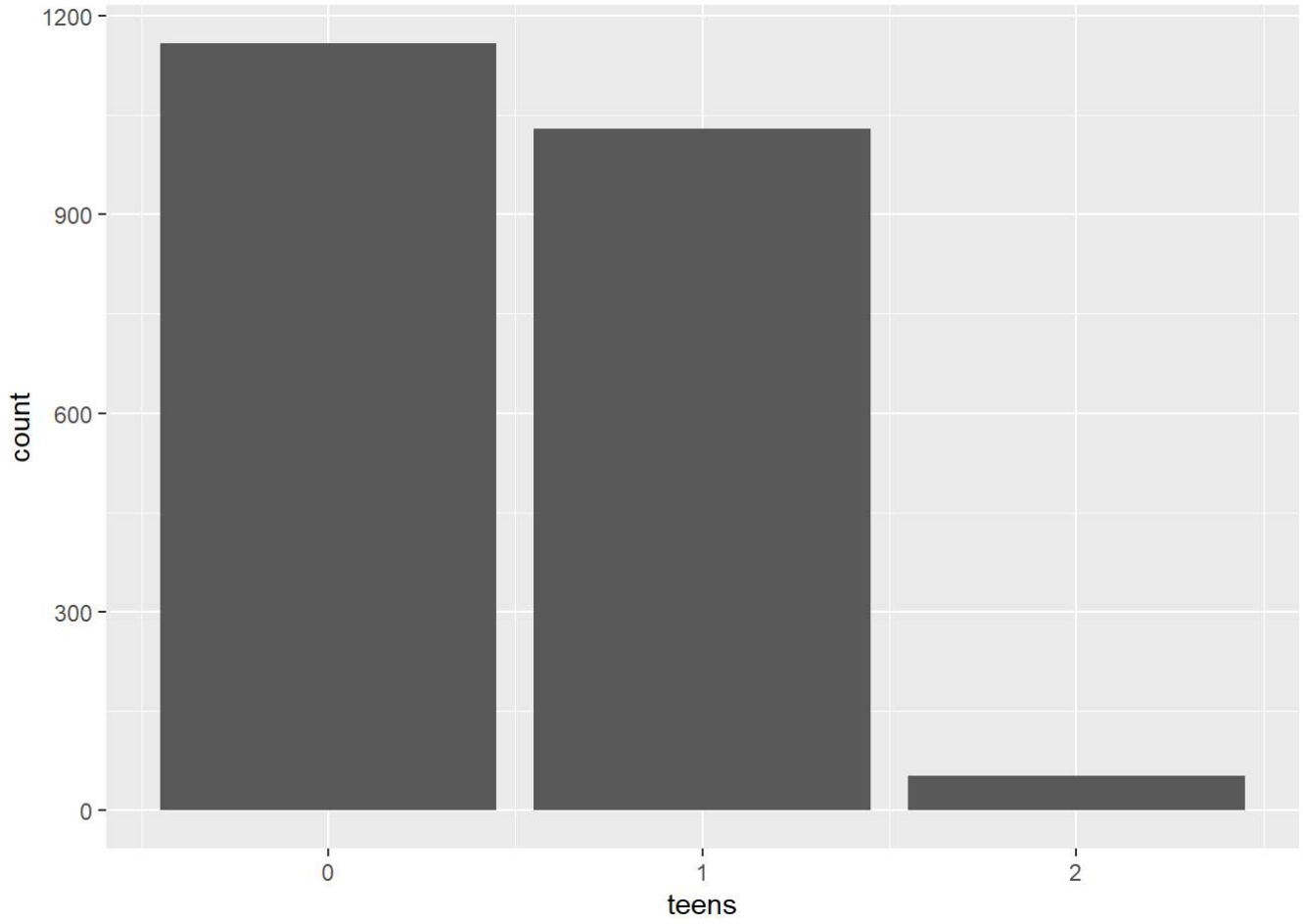

```
ggplot(retail,aes(kids))+geom_bar()
```



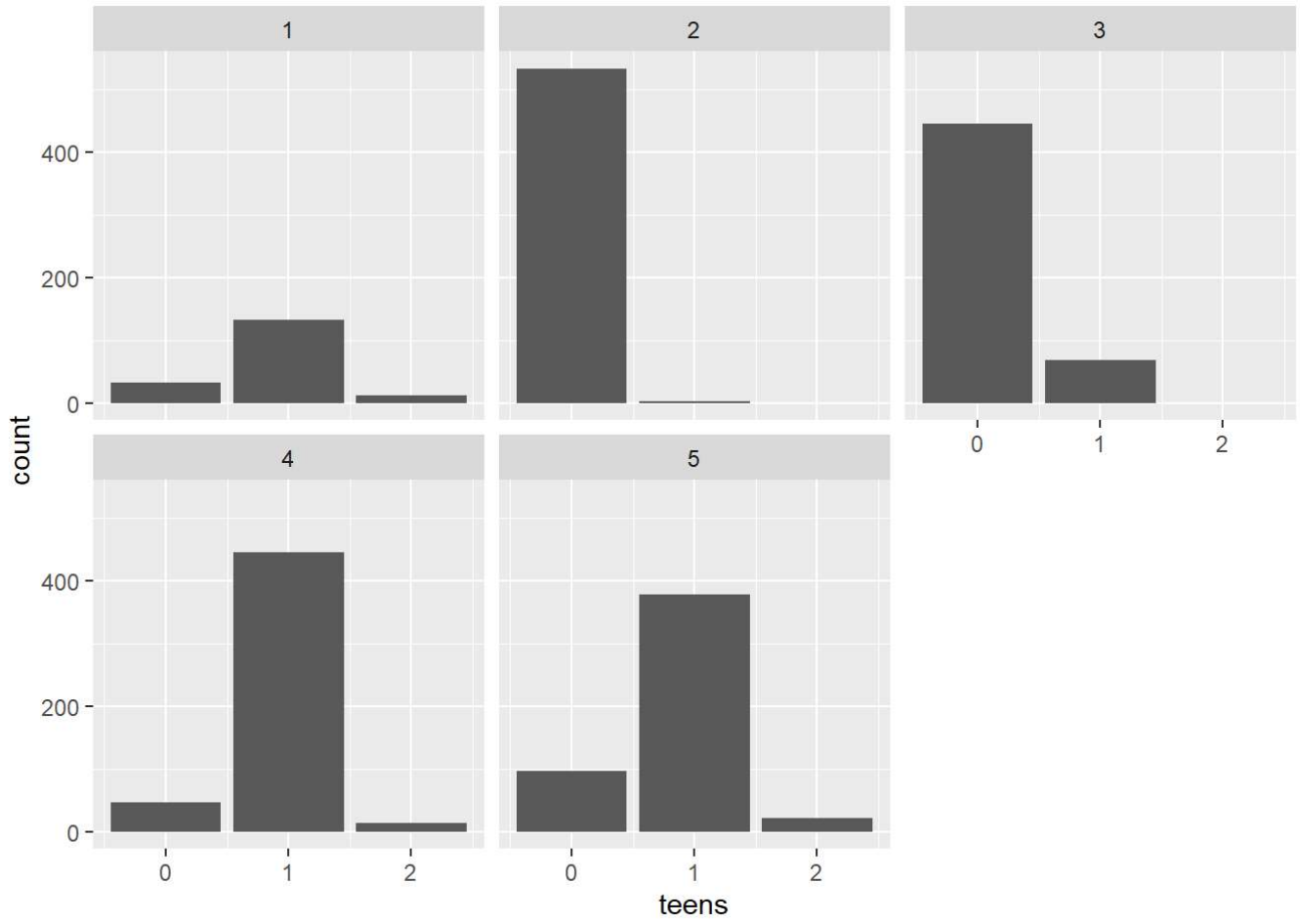

```
ggplot(retail,aes(kids))+geom_bar()+facet_wrap(~clusters5$cluster)
```



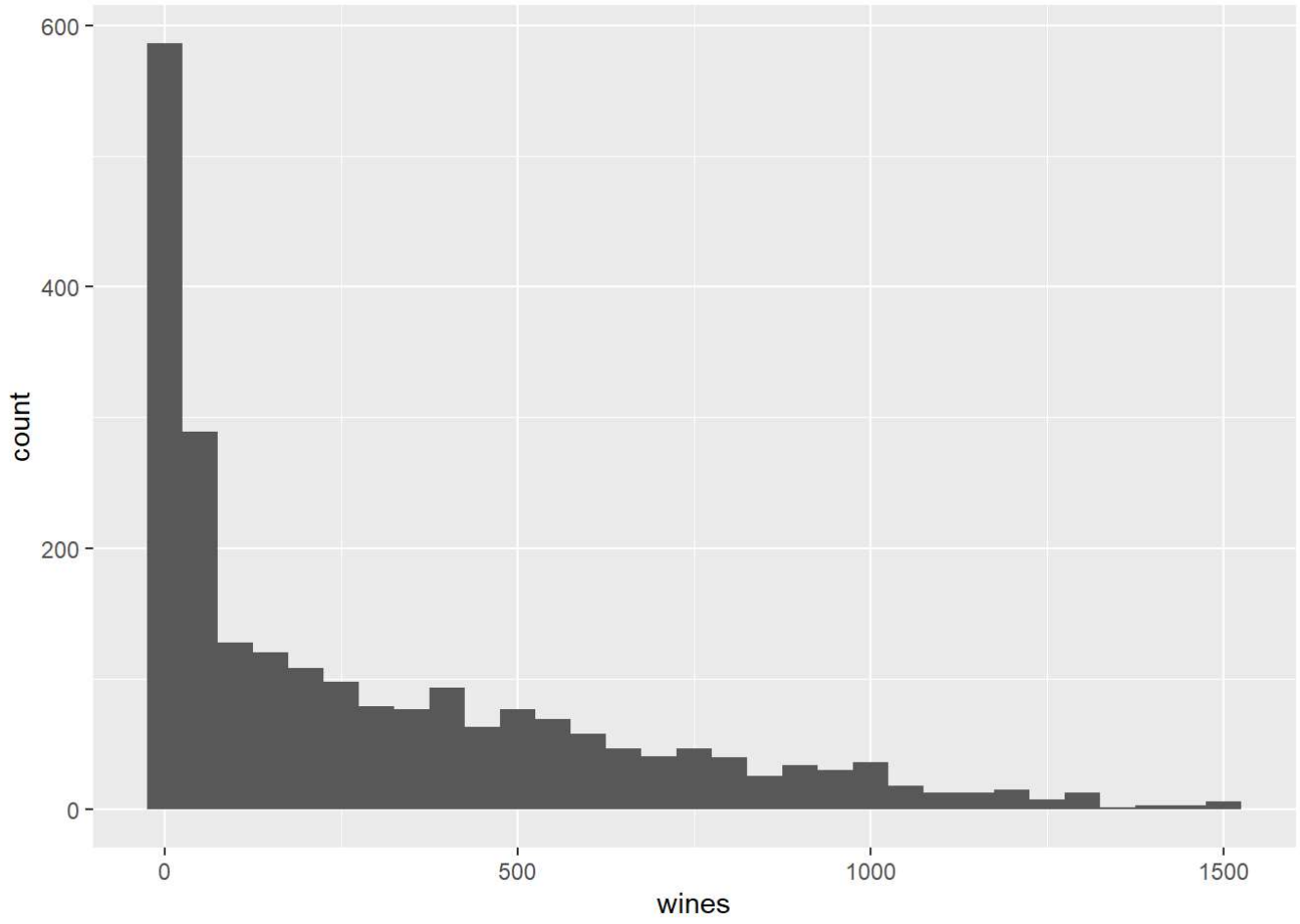
```
ggplot(retail,aes(teens))+geom_bar()
```



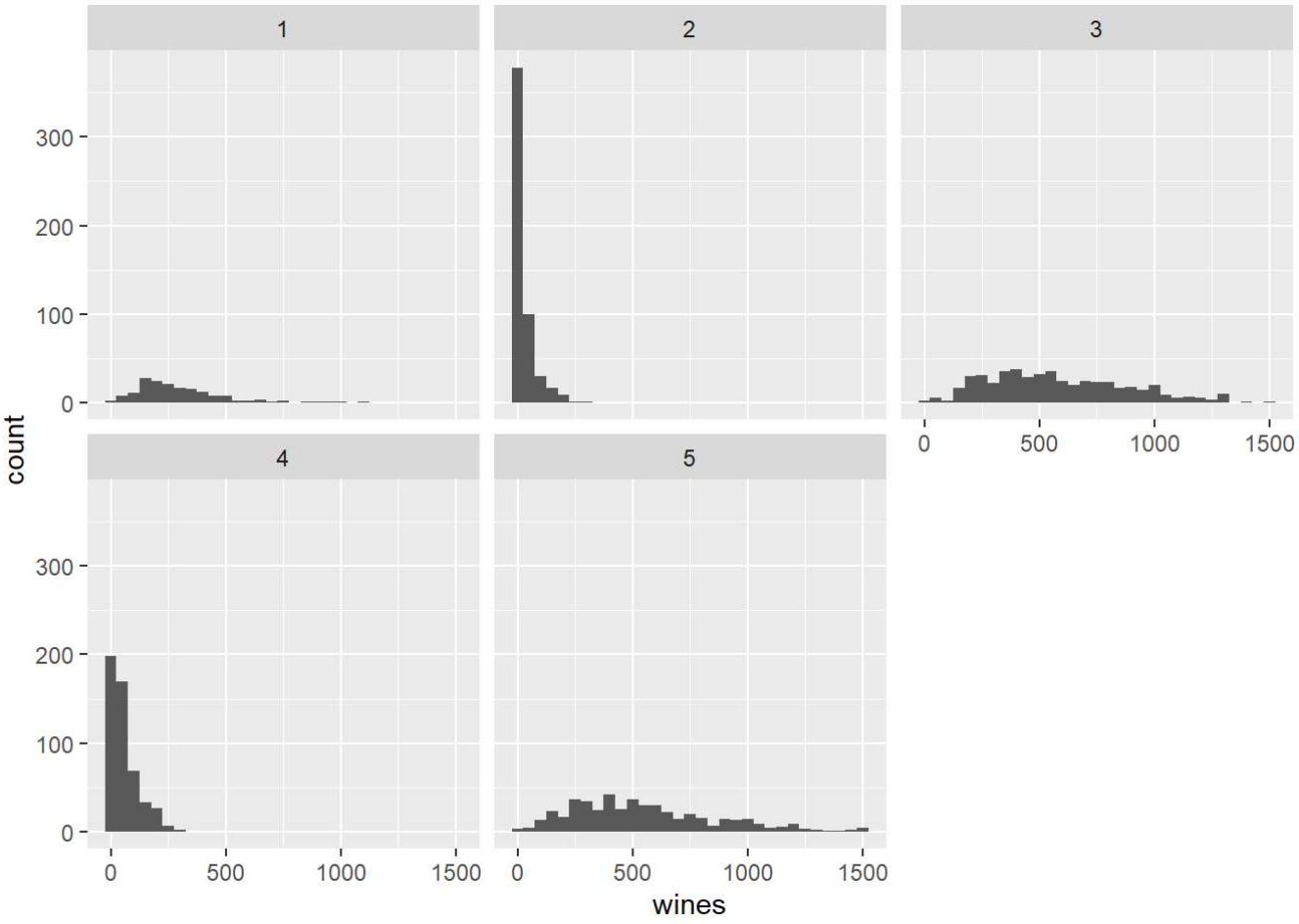

```
ggplot(retail,aes(teens))+geom_bar()+facet_wrap(~clusters5$cluster)
```



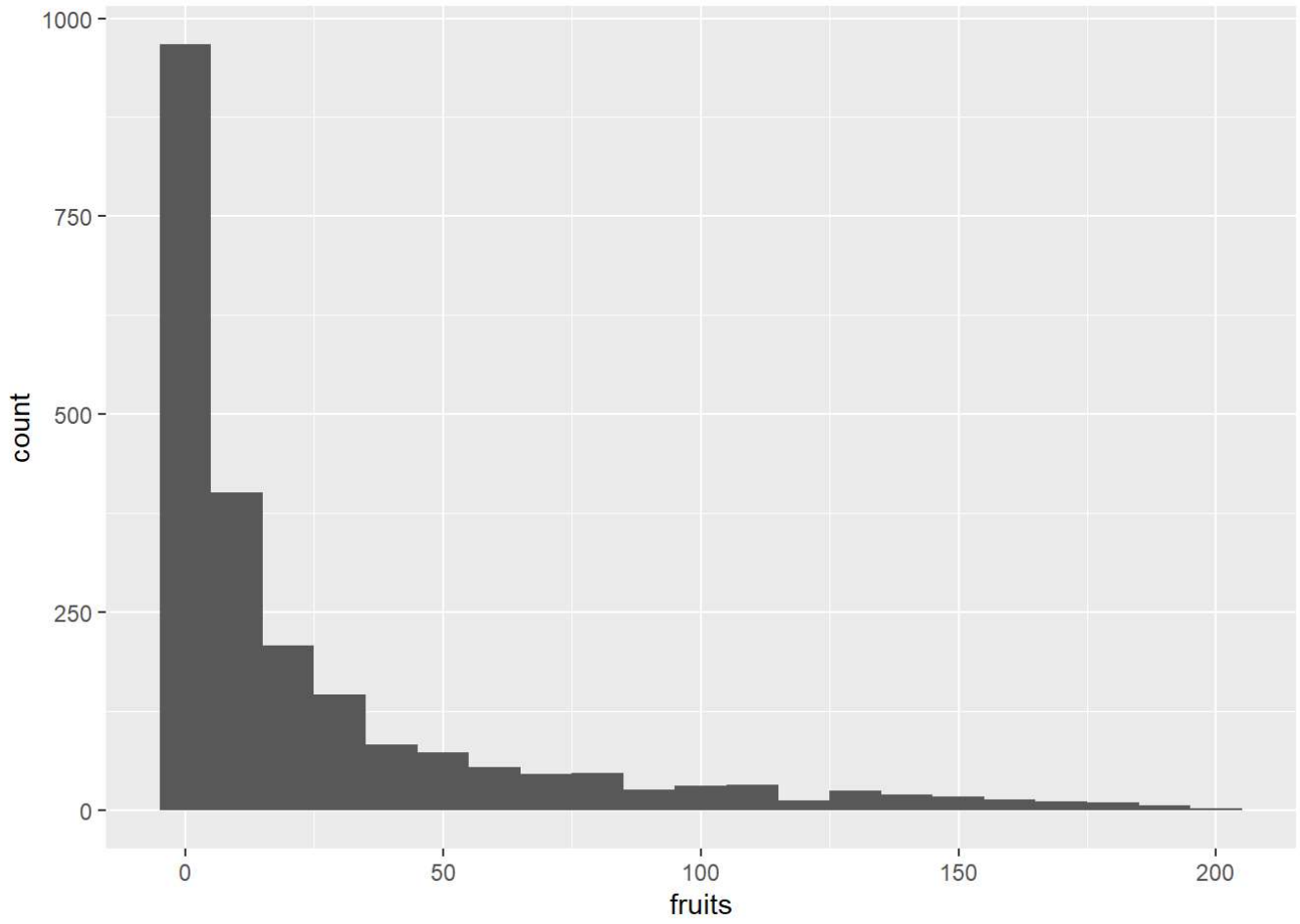

```
ggplot(retail,aes(wines))+geom_histogram(binwidth=50)
```



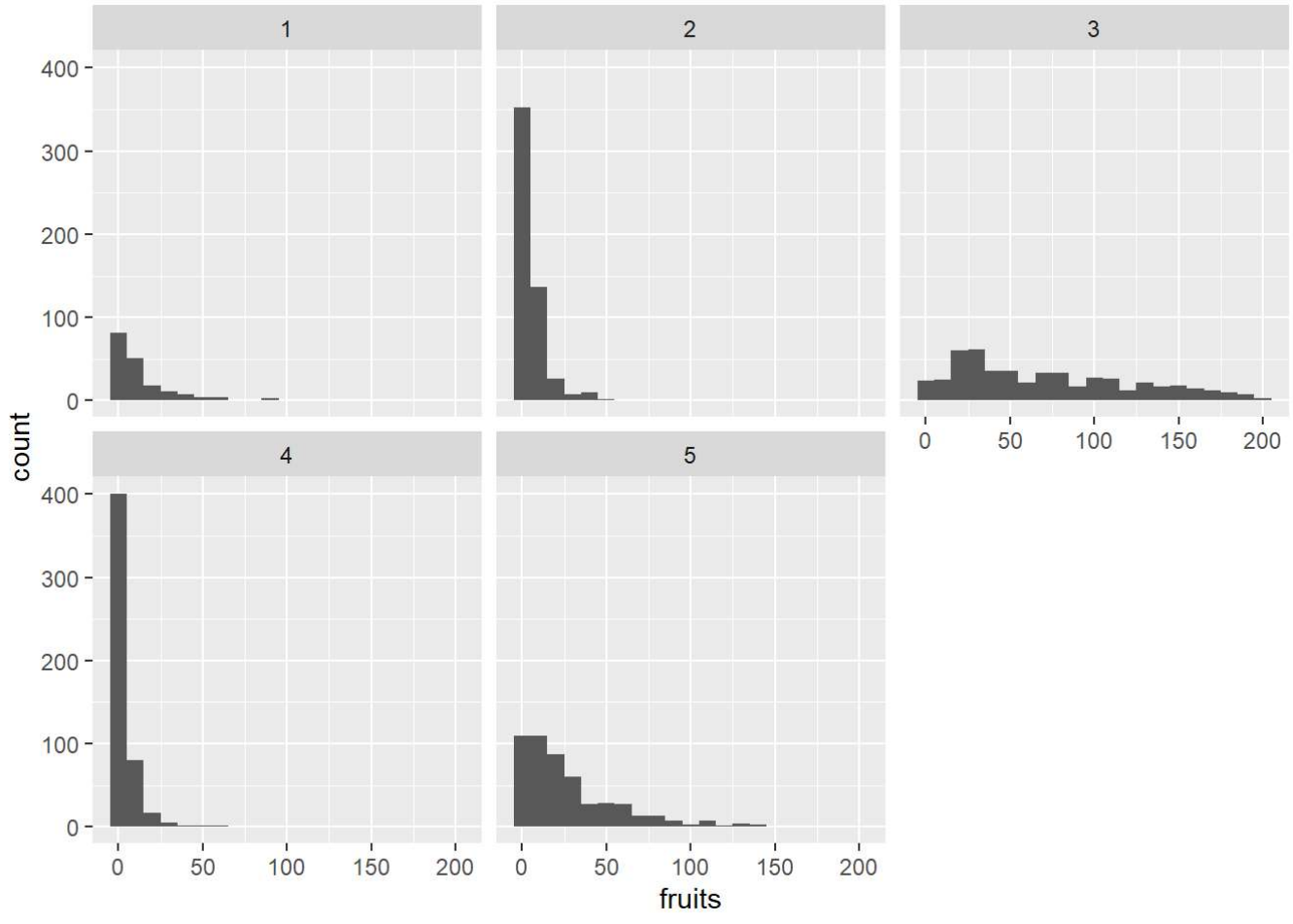
```
ggplot(retail,aes(wines))+geom_histogram(binwidth=50)+facet_wrap(~clusters5$cluster)
```



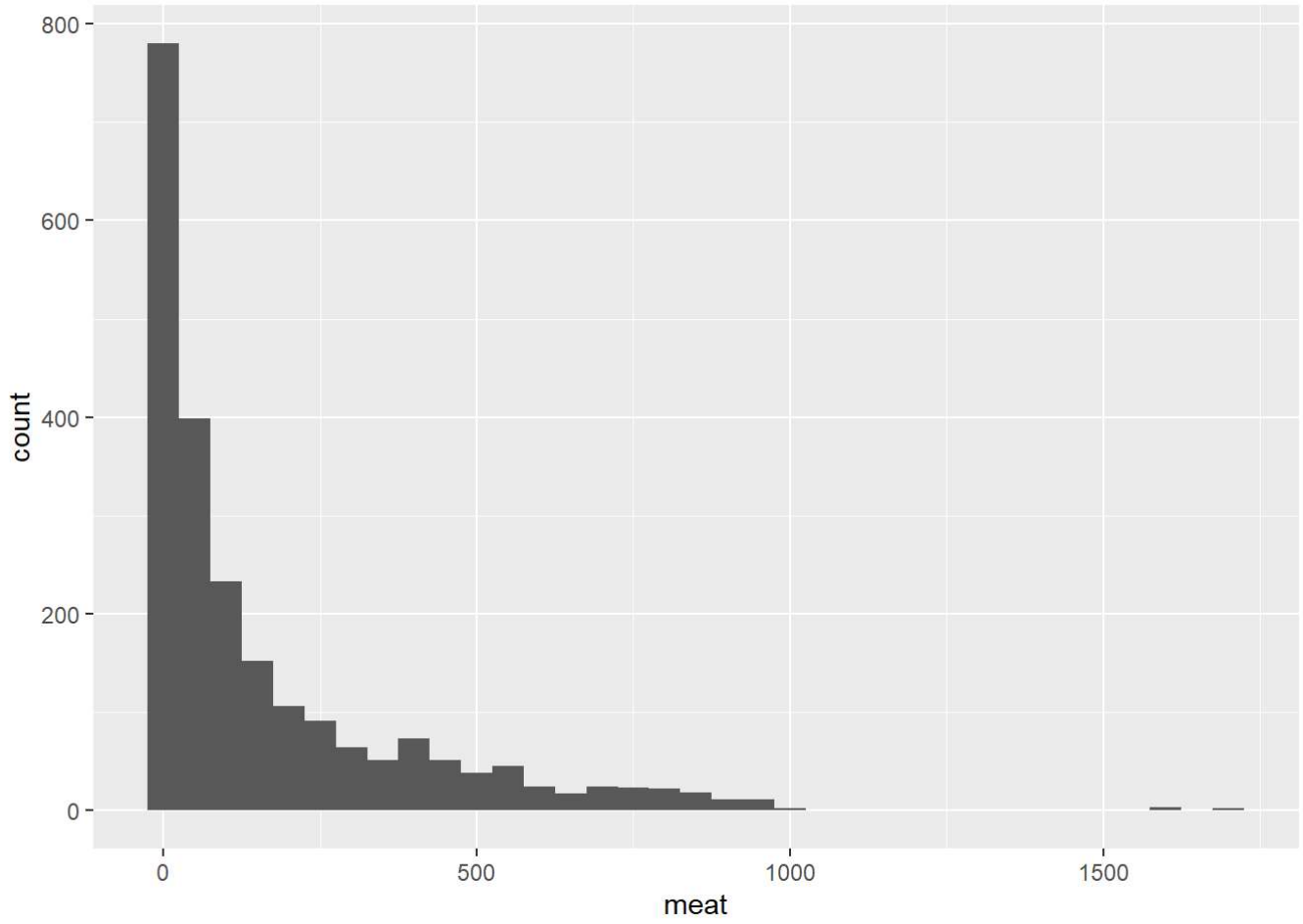

```
ggplot(retail,aes(fruits))+geom_histogram(binwidth=10)
```



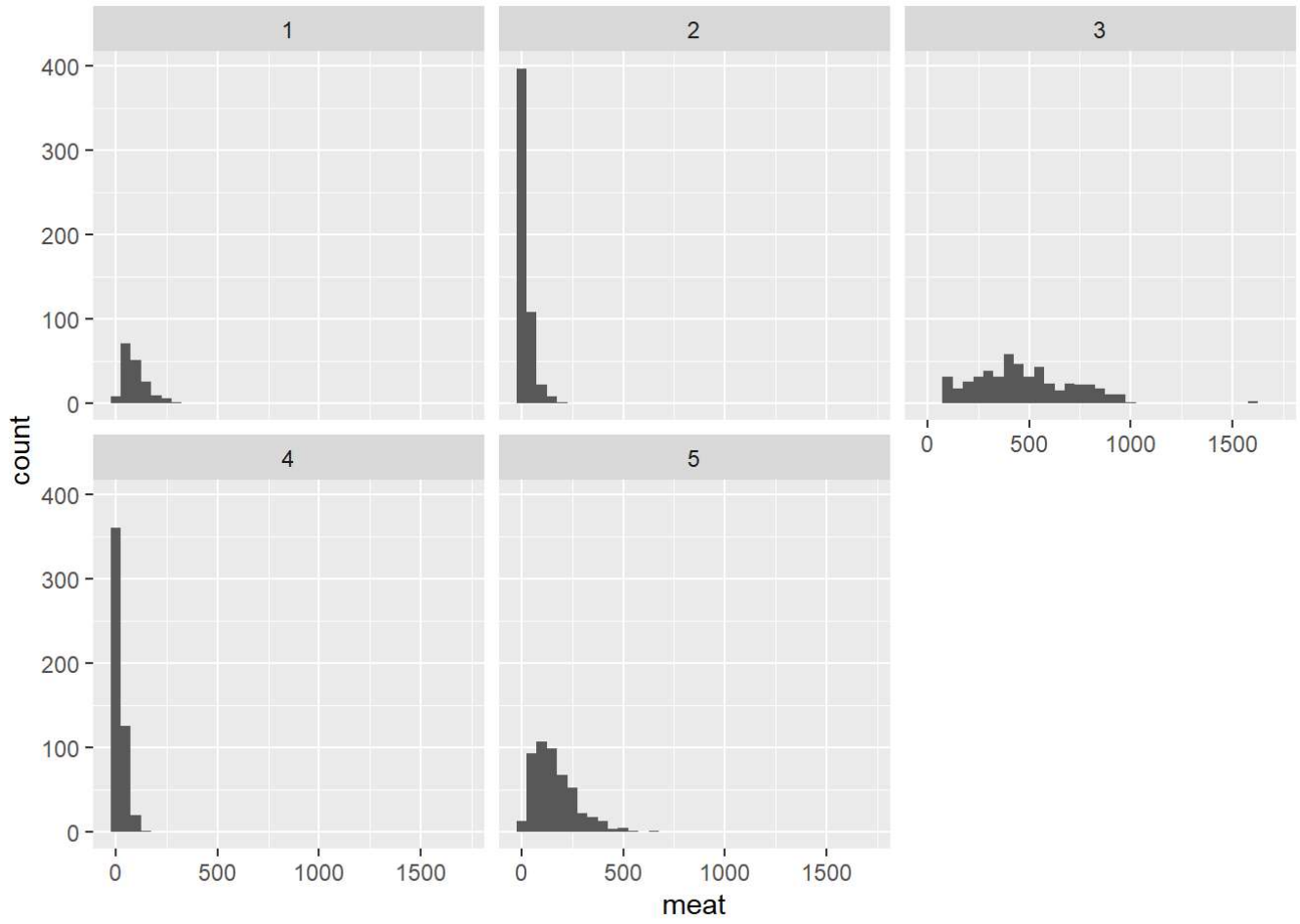

```
ggplot(retail,aes(fruits))+geom_histogram(binwidth=10)+facet_wrap(~clusters5$cluster)
```



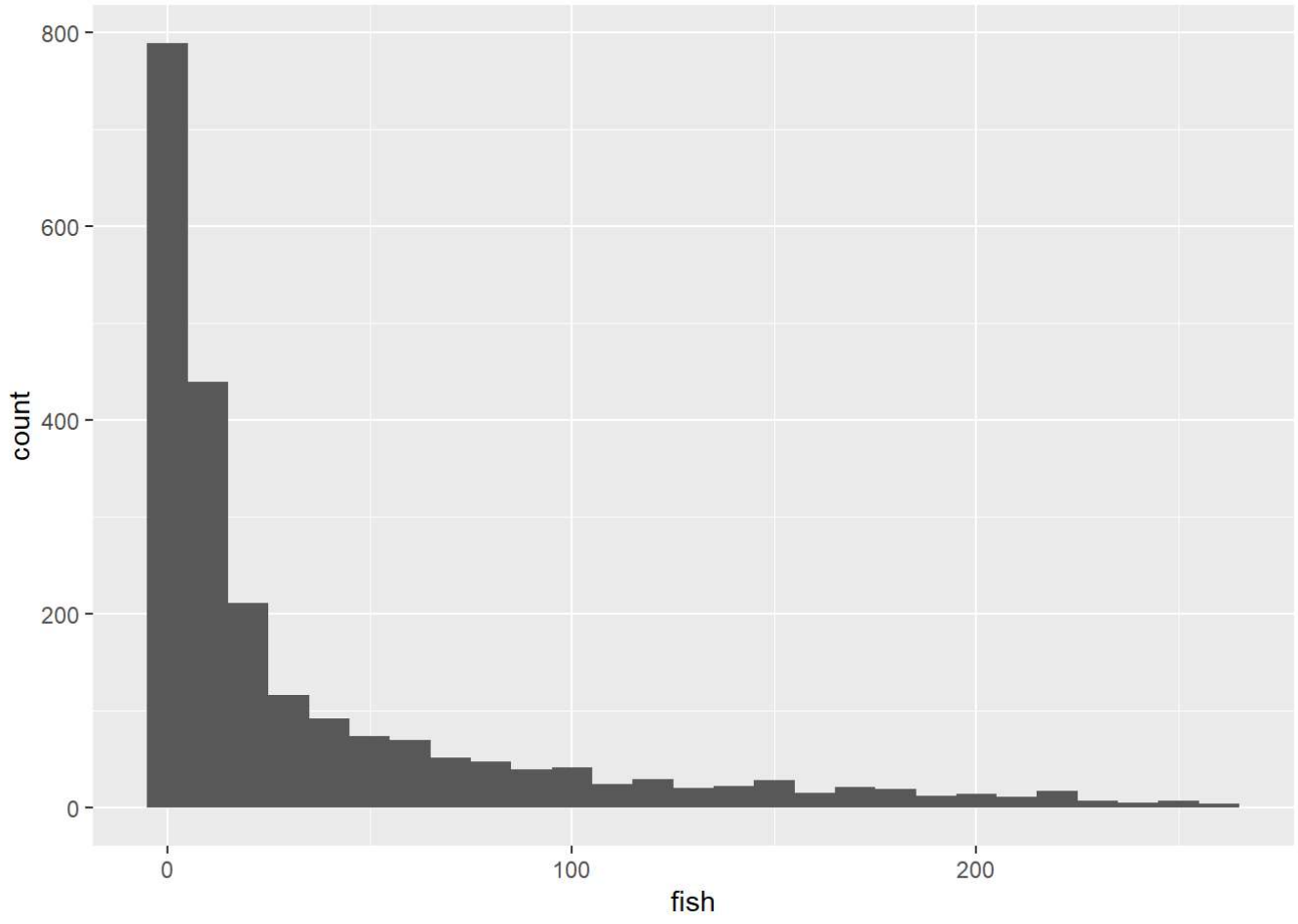
```
ggplot(retail,aes(meat))+geom_histogram(binwidth=50)
```



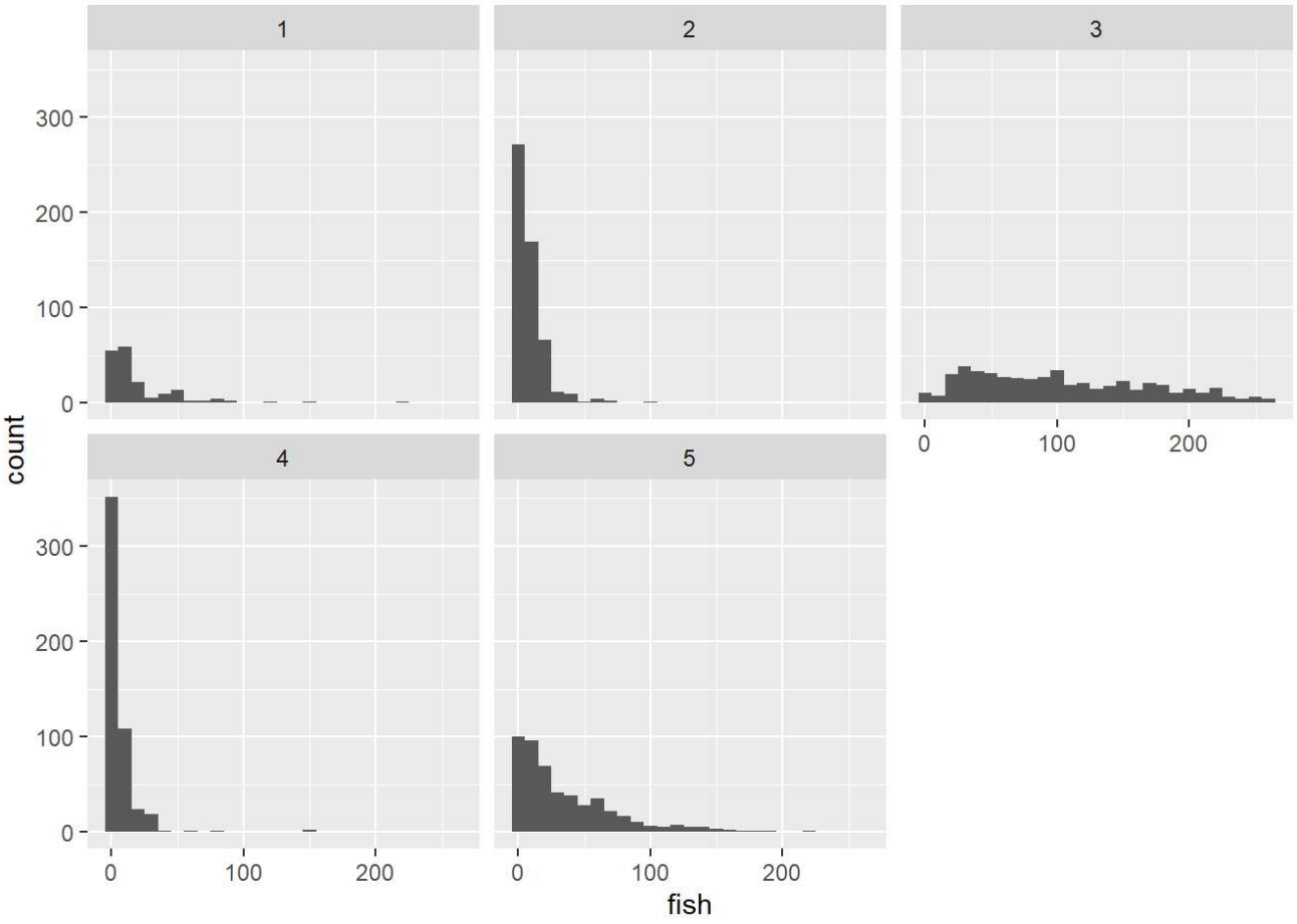

```
ggplot(retail,aes(meat))+geom_histogram(binwidth=50)+facet_wrap(~clusters5$cluster)
```



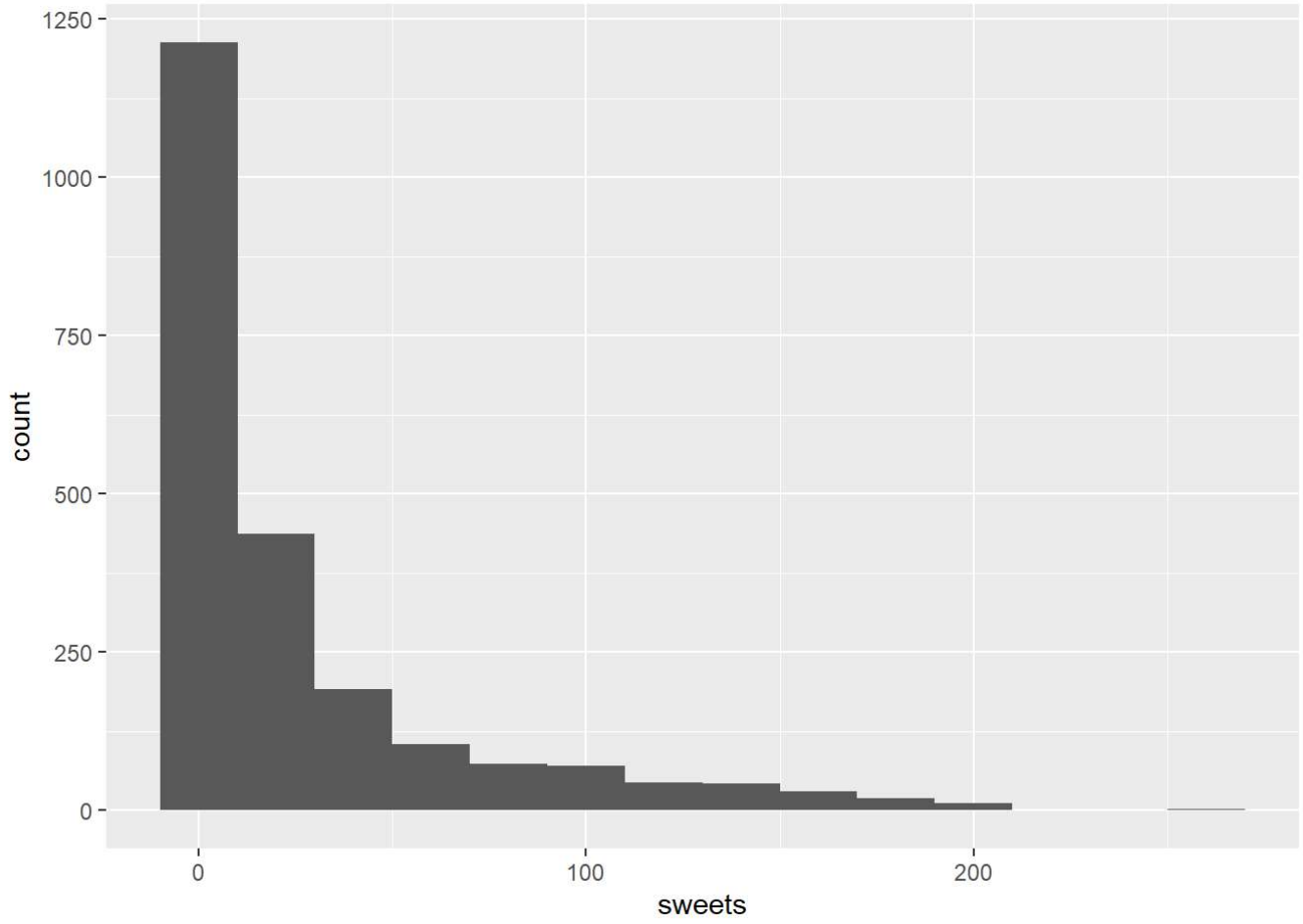

```
ggplot(retail,aes(fish))+geom_histogram(binwidth=10)
```



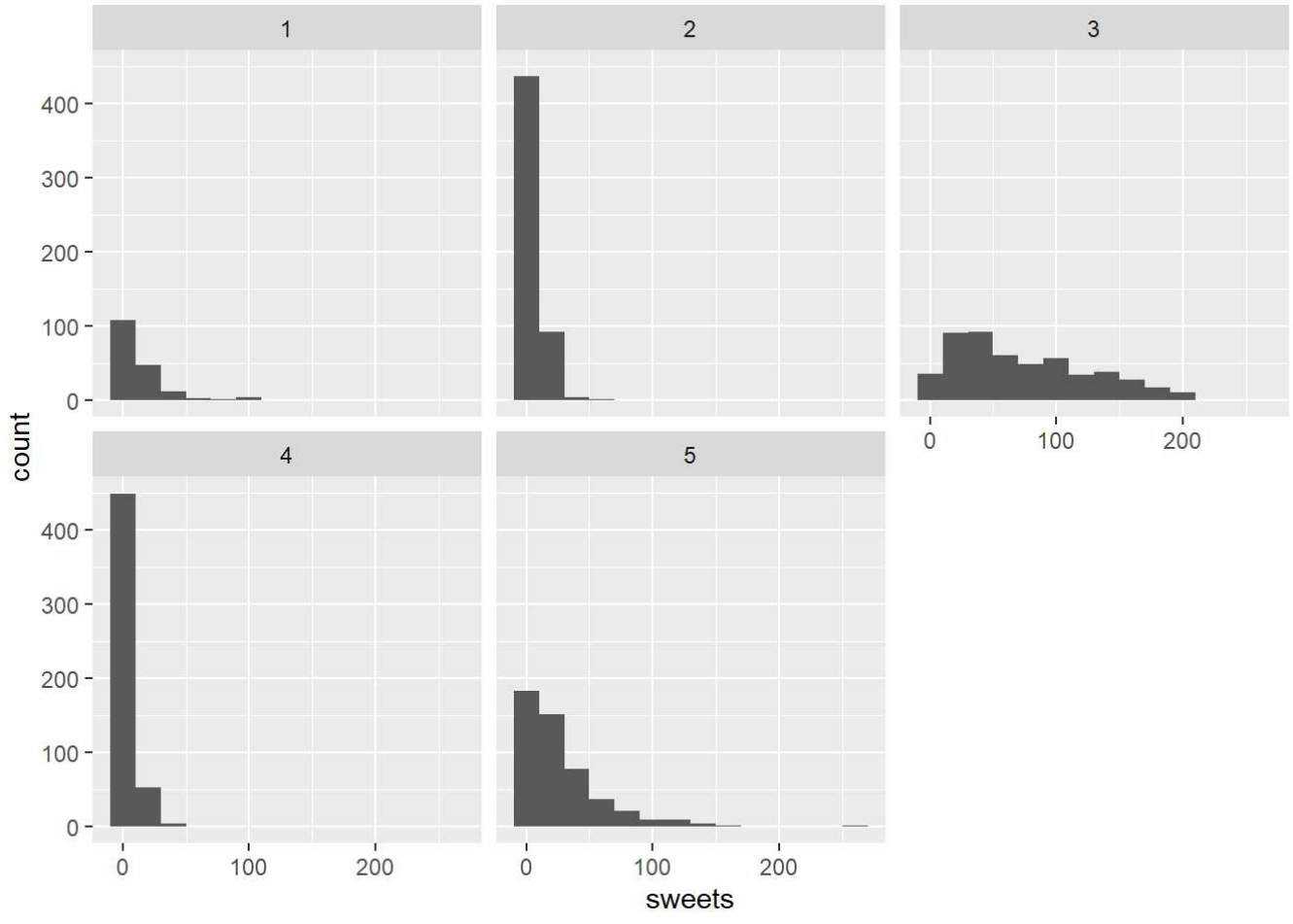
```
ggplot(retail,aes(fish))+geom_histogram(binwidth=10)+facet_wrap(~clusters5$cluster)
```



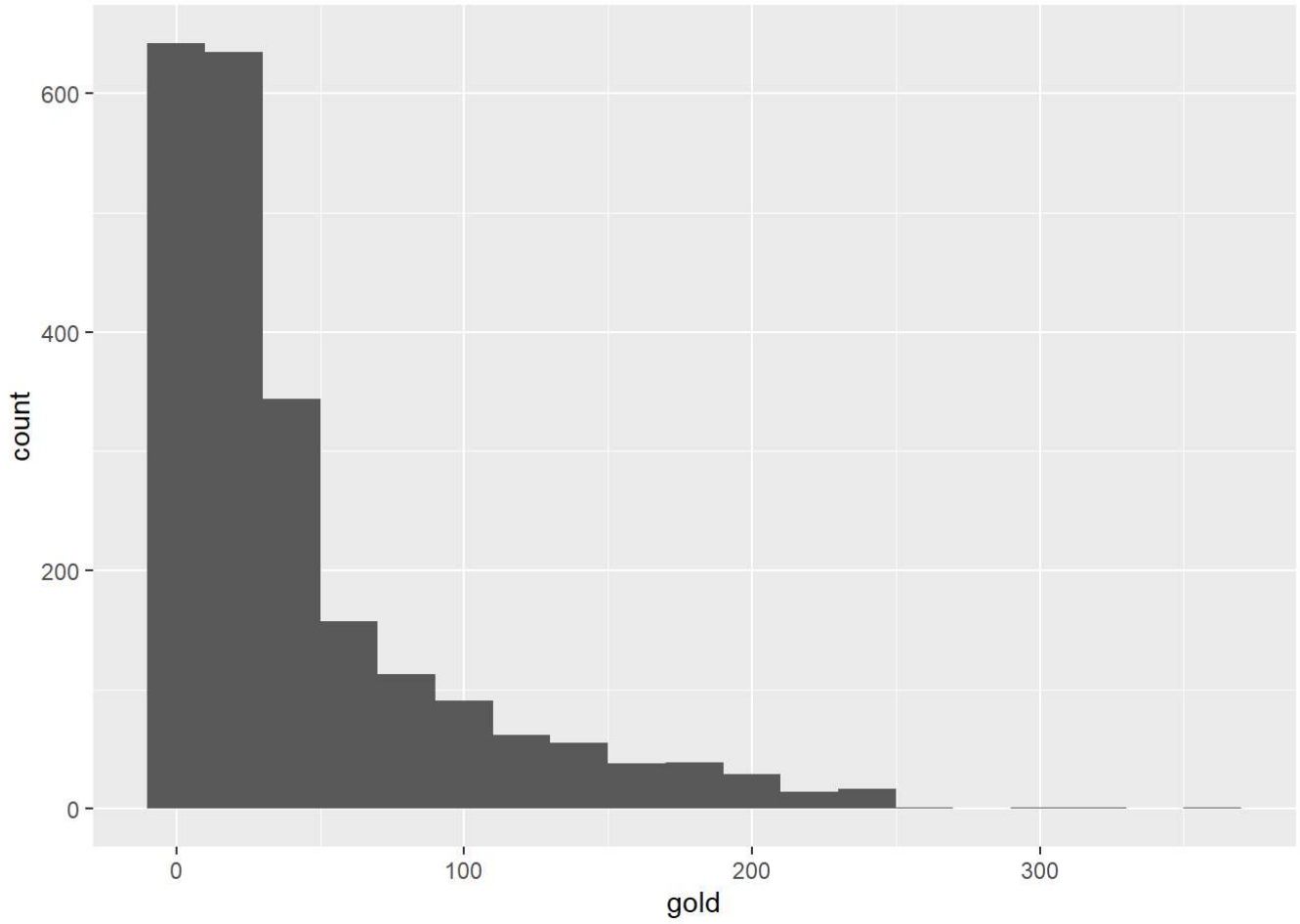

```
ggplot(retail,aes(sweets))+geom_histogram(binwidth=20)
```



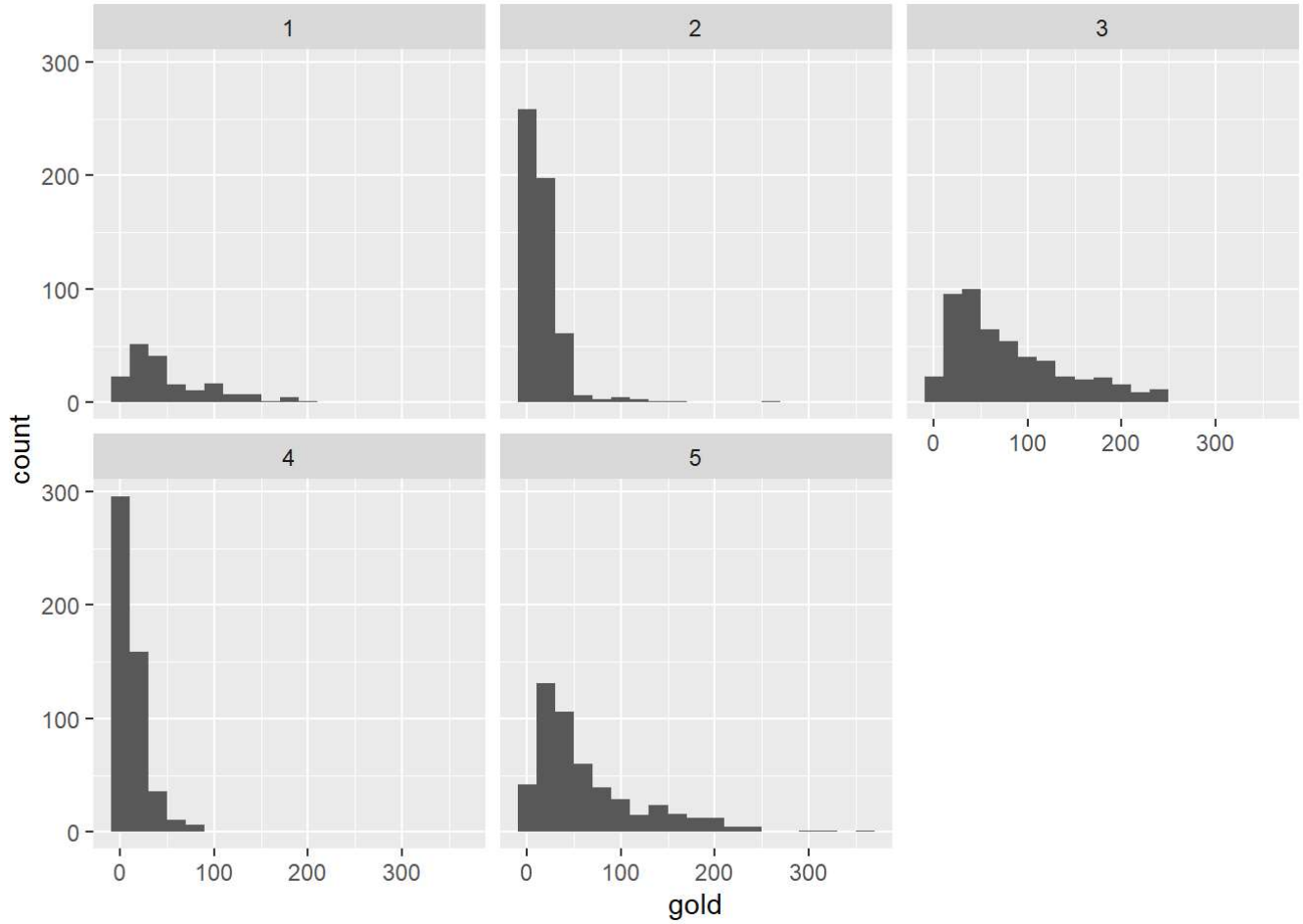

```
ggplot(retail,aes(sweets))+geom_histogram(binwidth=20)+facet_wrap(~clusters5$cluster)
```



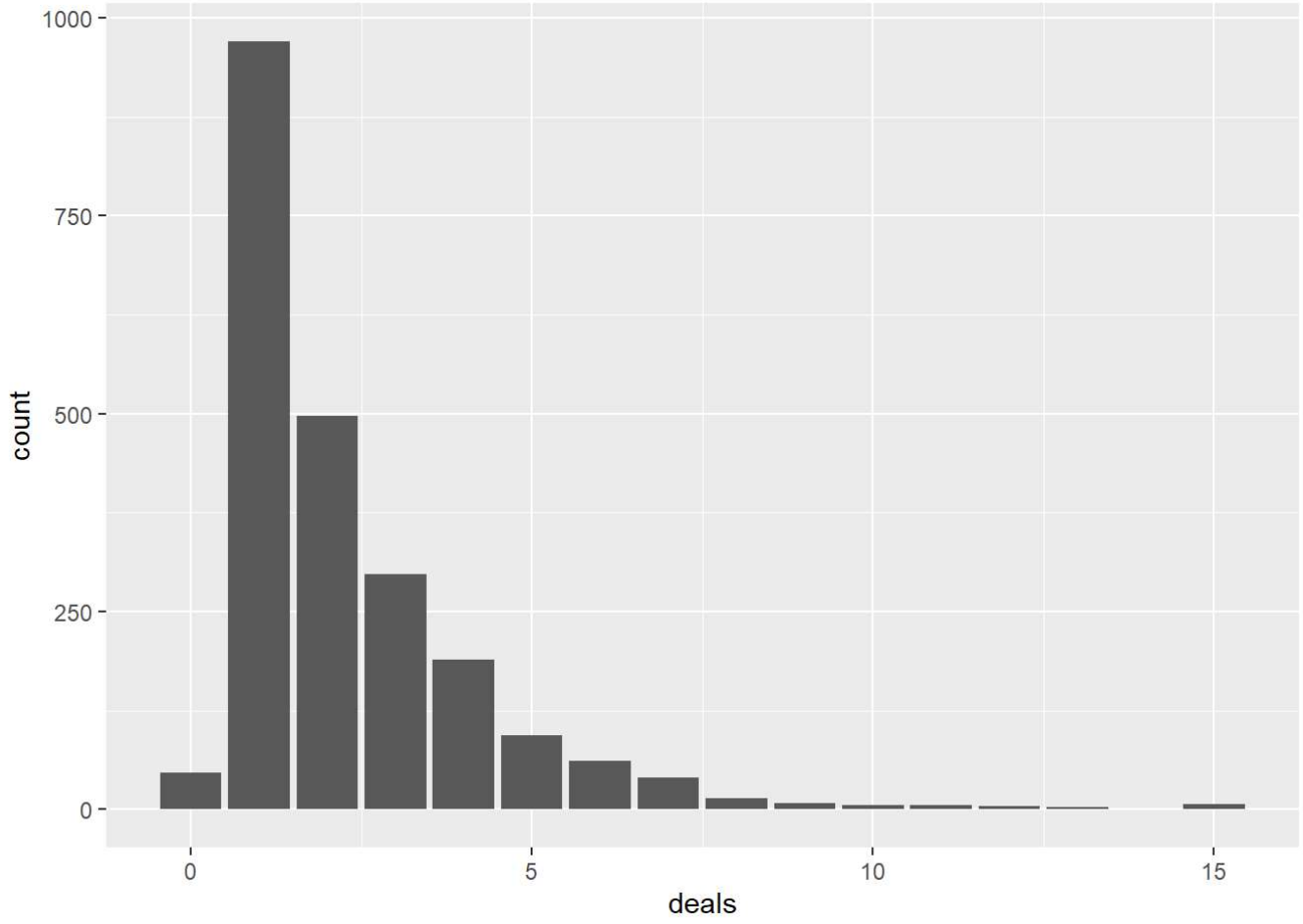
```
ggplot(retail,aes(gold))+geom_histogram(binwidth=20)
```



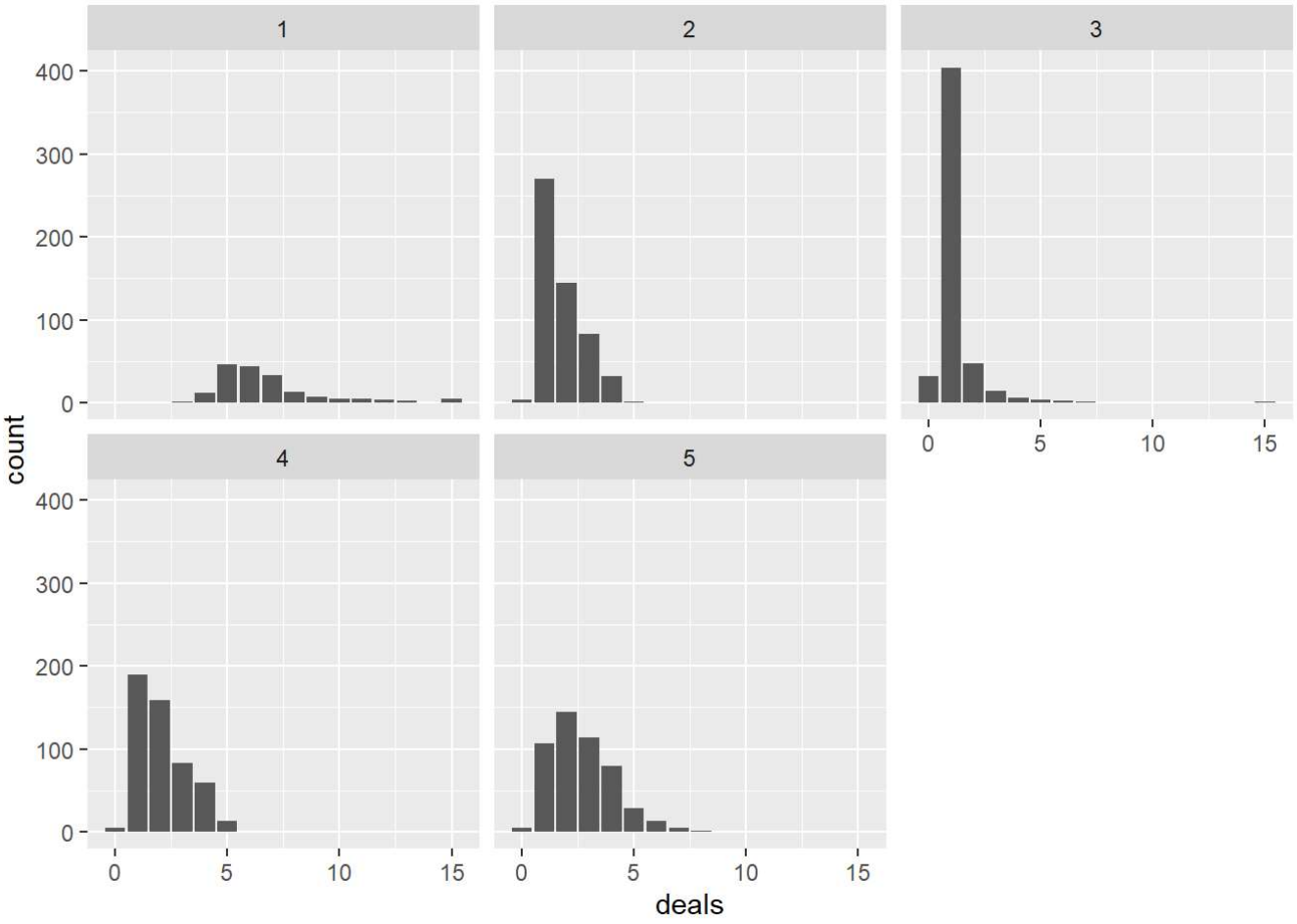

```
ggplot(retail,aes(gold))+geom_histogram(binwidth=20)+facet_wrap(~clusters5$cluster)
```



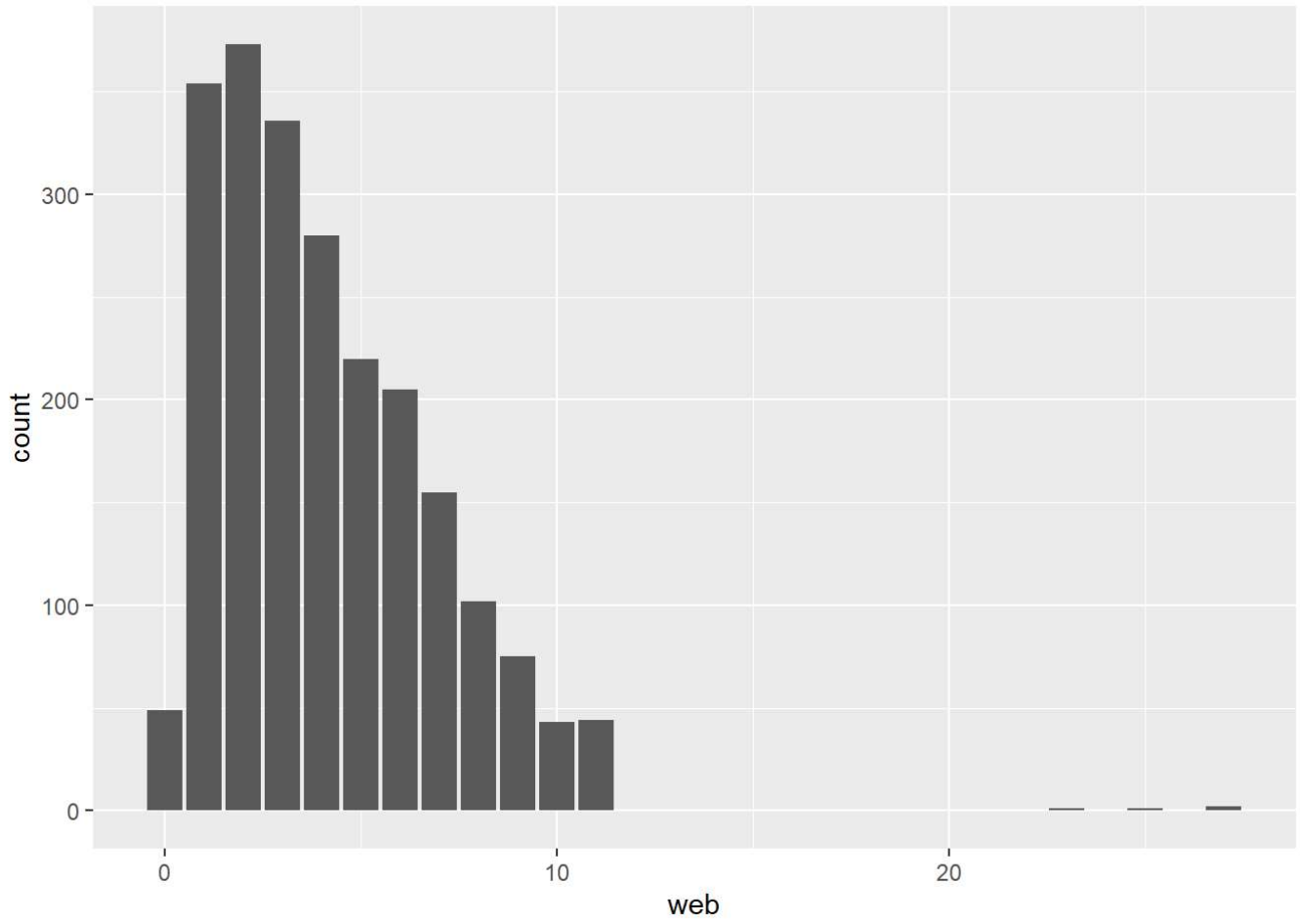

```
ggplot(retail,aes(deals))+geom_bar()
```



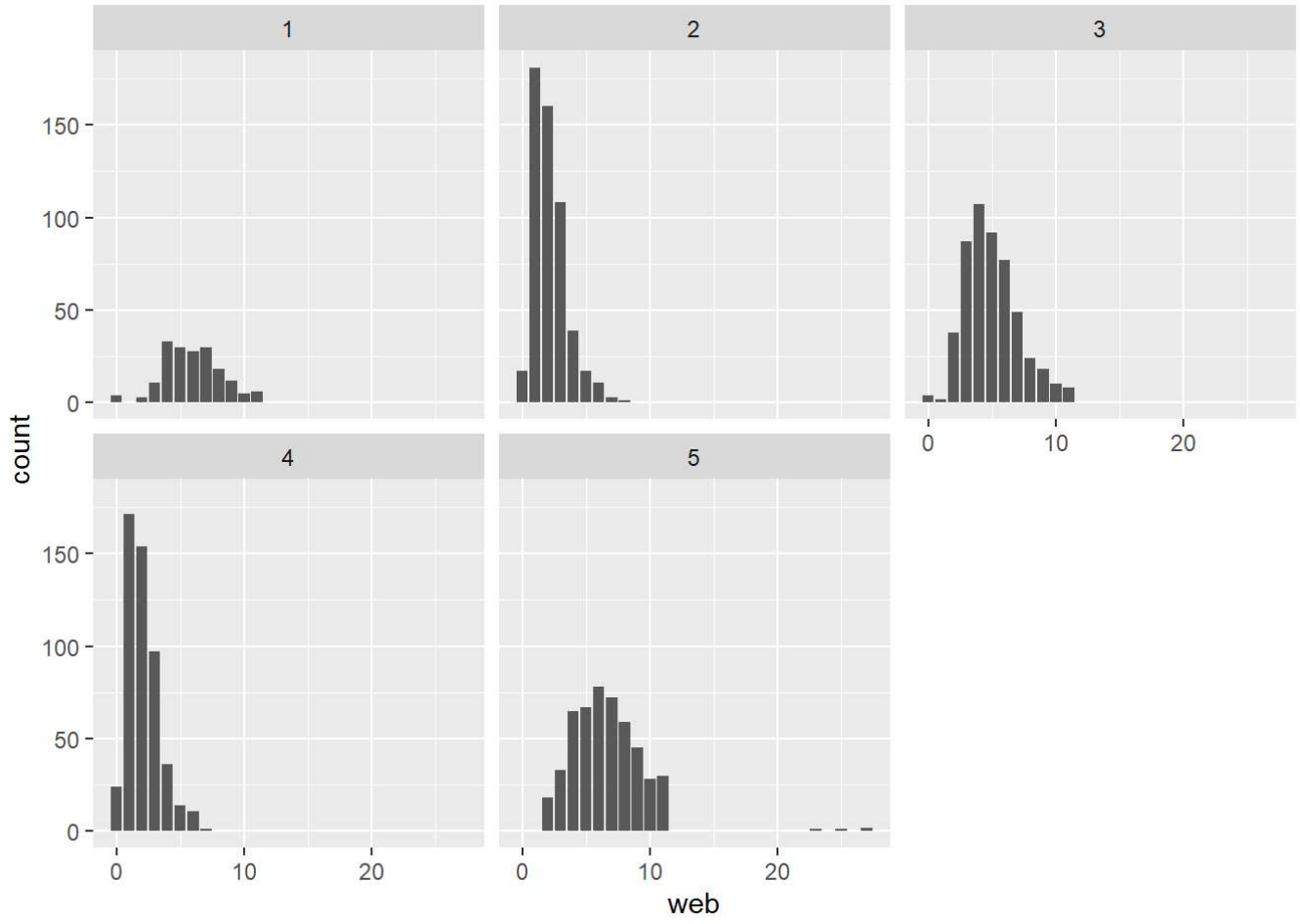
```
ggplot(retail,aes(deals))+geom_bar()+facet_wrap(~clusters5$cluster)
```



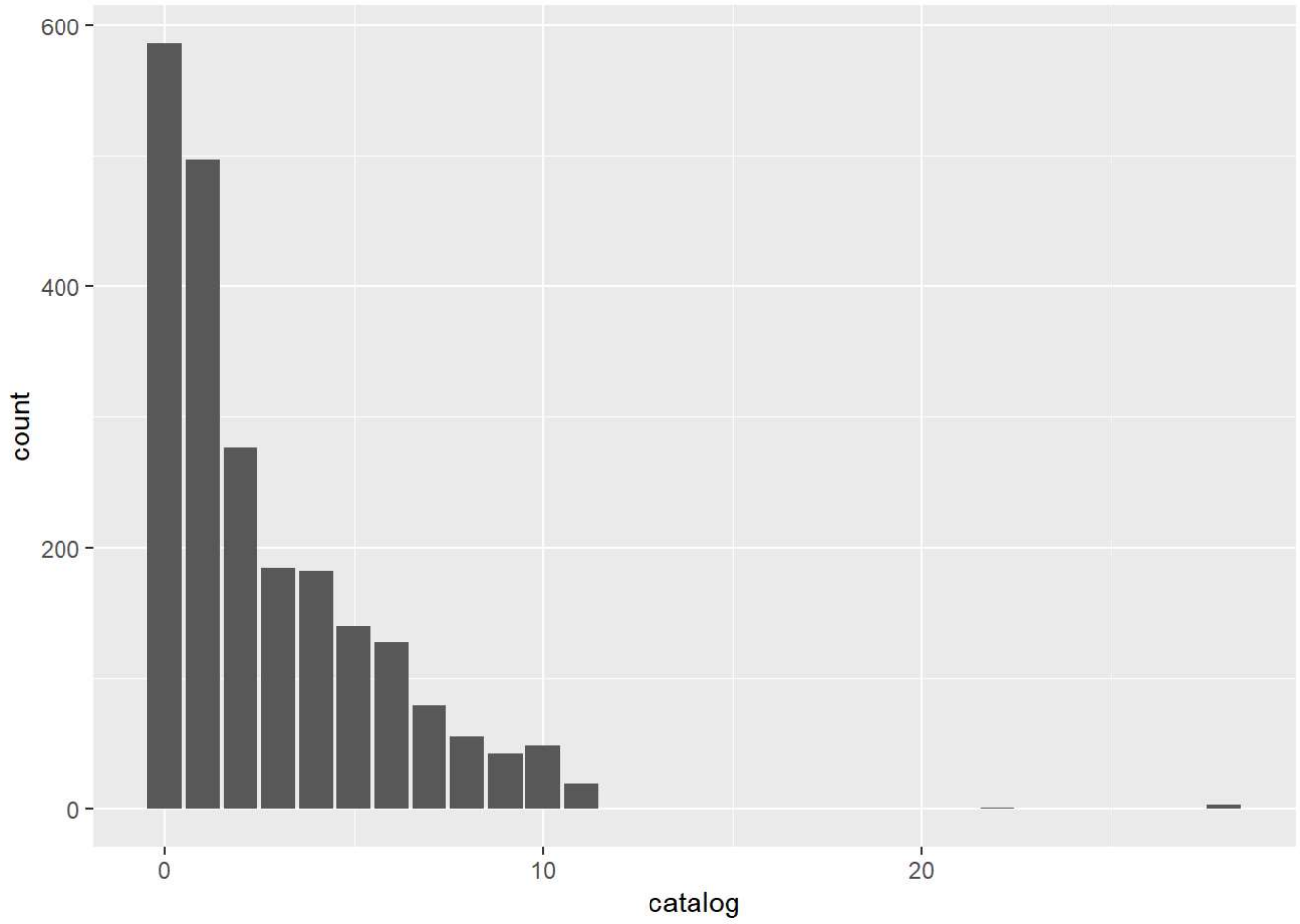

```
ggplot(retail,aes(web))+geom_bar()
```



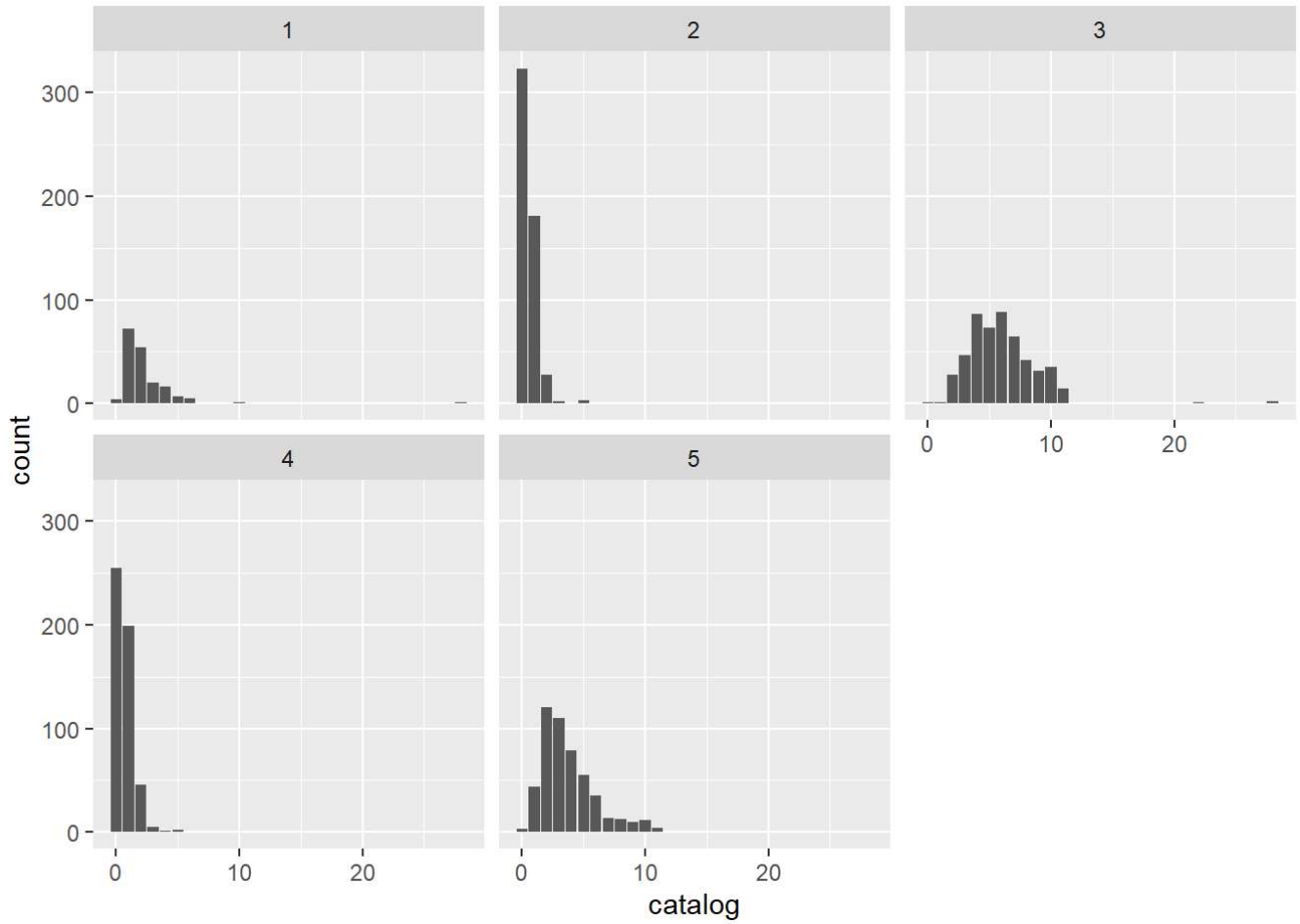

```
ggplot(retail,aes(web))+geom_bar()+facet_wrap(~clusters5$cluster)
```



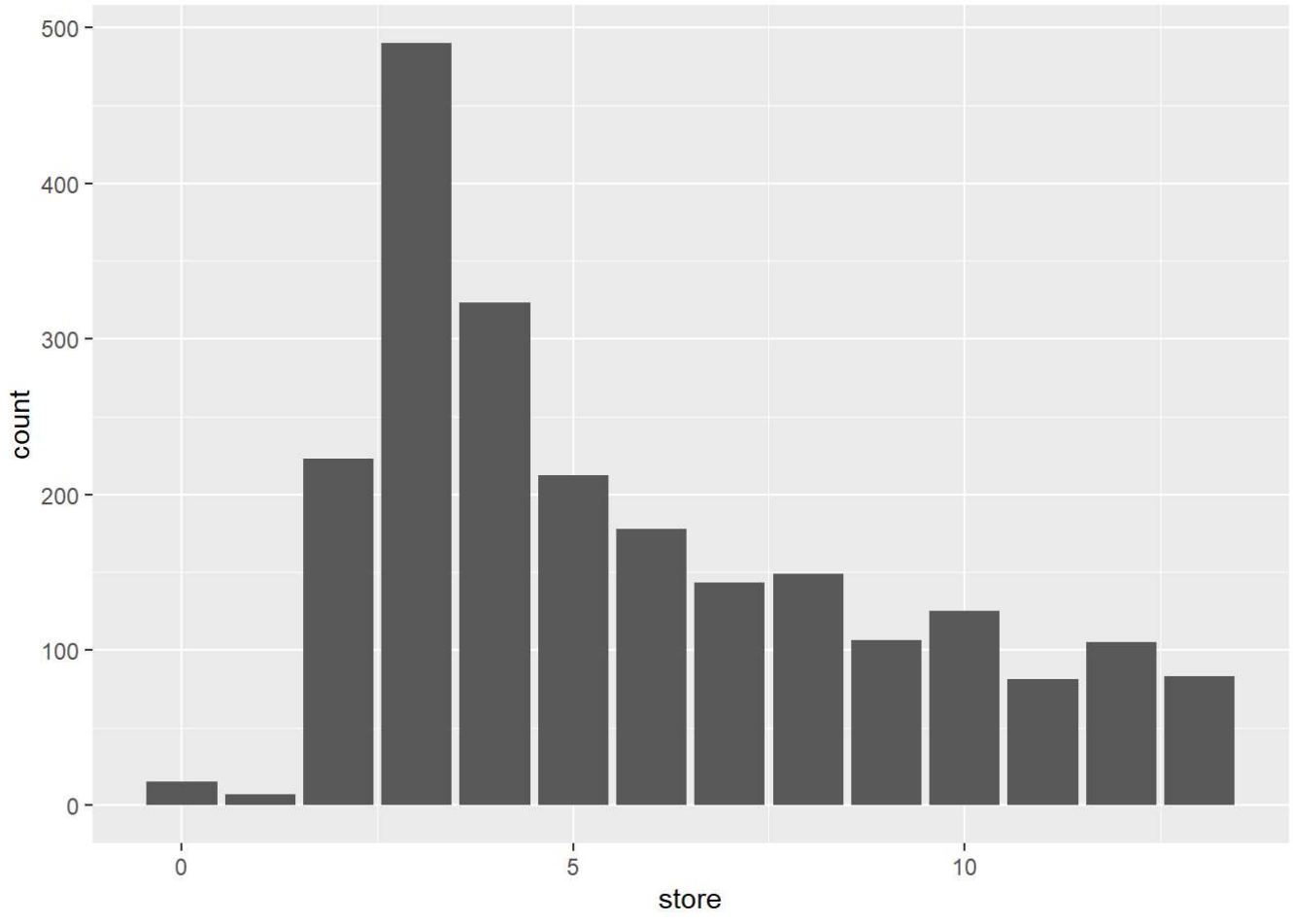
```
ggplot(retail,aes(catalog))+geom_bar()
```



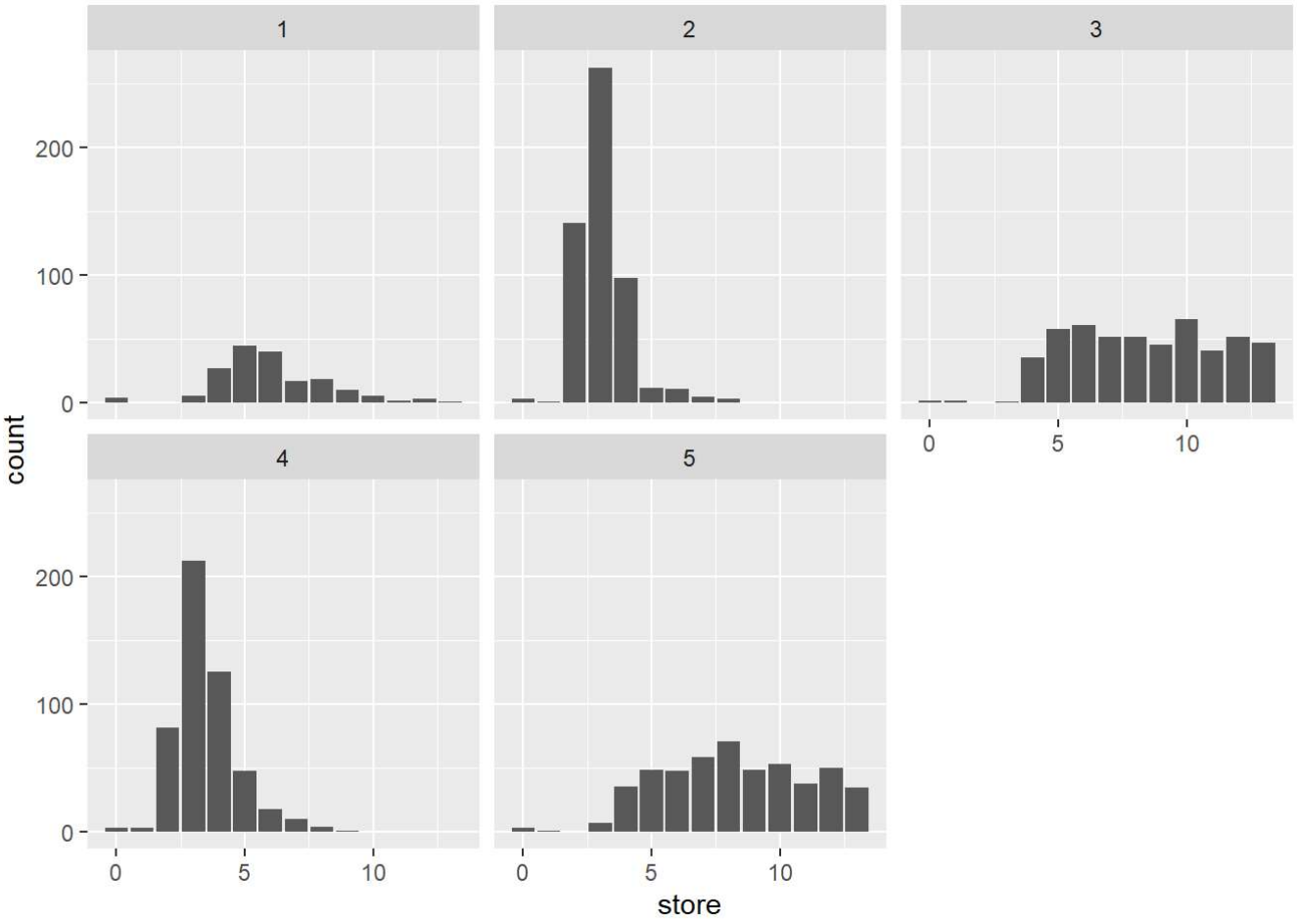

```
ggplot(retail,aes(catalog))+geom_bar()+facet_wrap(~clusters5$cluster)
```



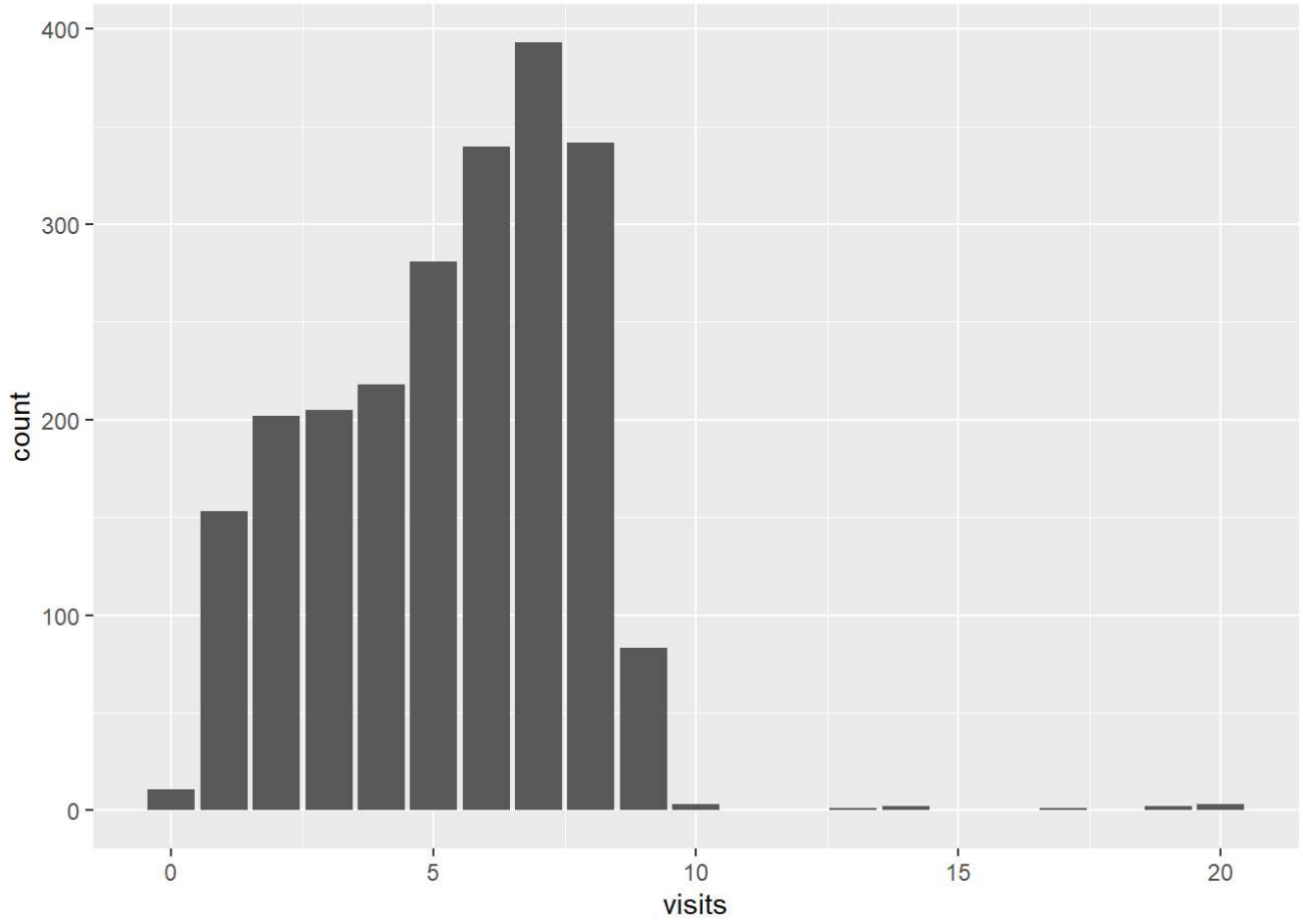

```
ggplot(retail,aes(store))+geom_bar()
```



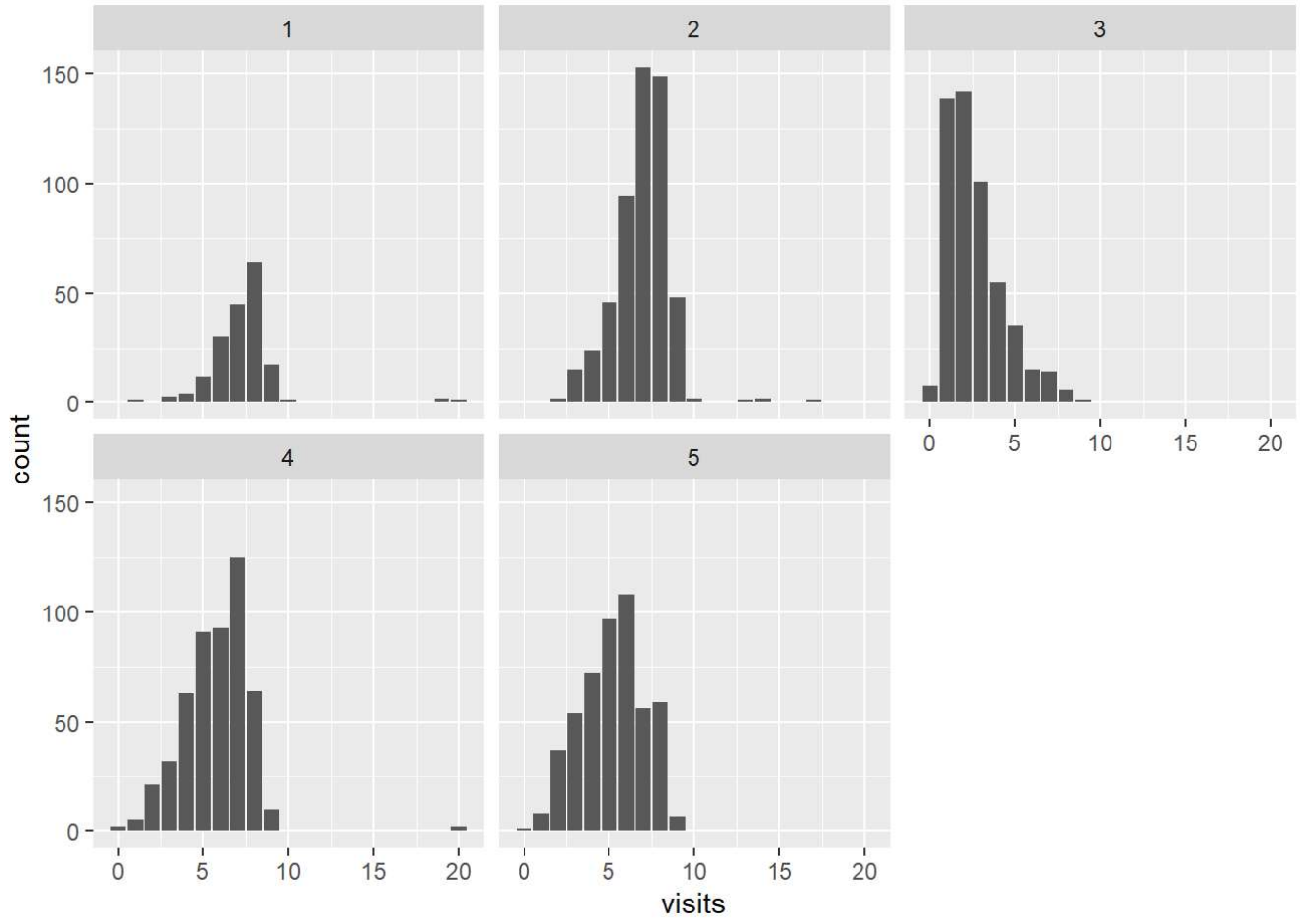
```
ggplot(retail,aes(store))+geom_bar()+facet_wrap(~clusters5$cluster)
```



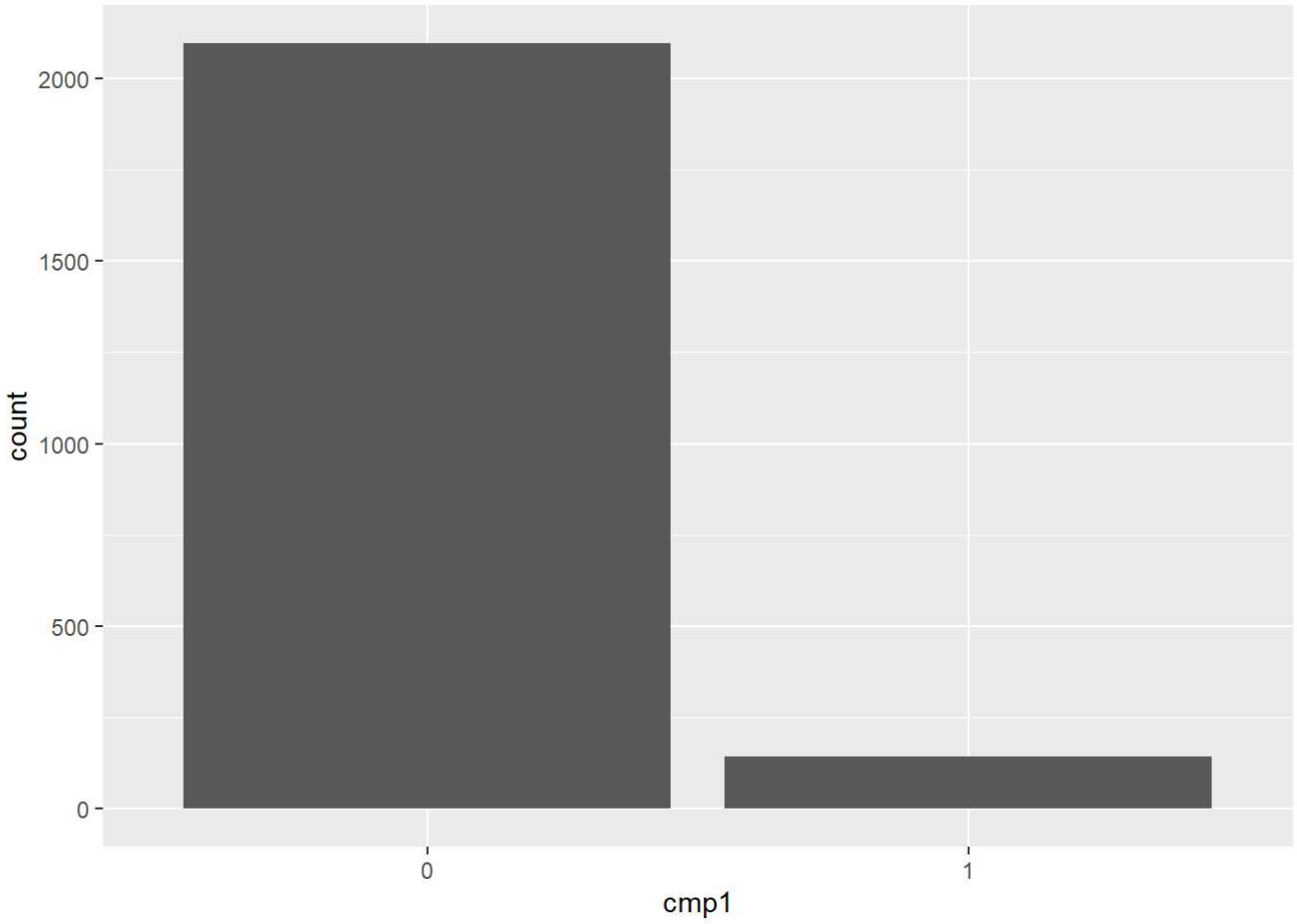

```
ggplot(retail,aes(visits))+geom_bar()
```



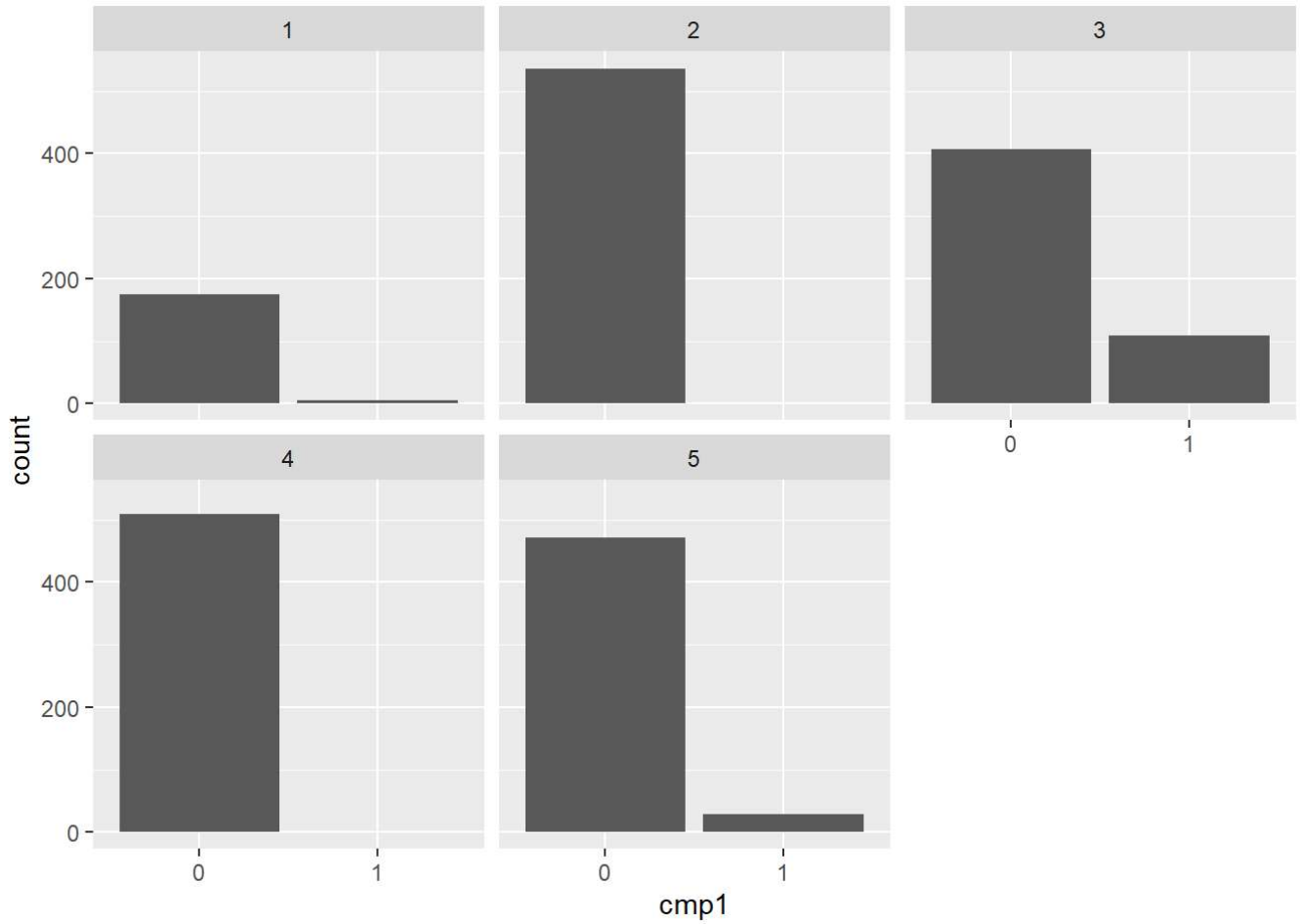

```
ggplot(retail,aes(visits))+geom_bar()+facet_wrap(~clusters5$cluster)
```



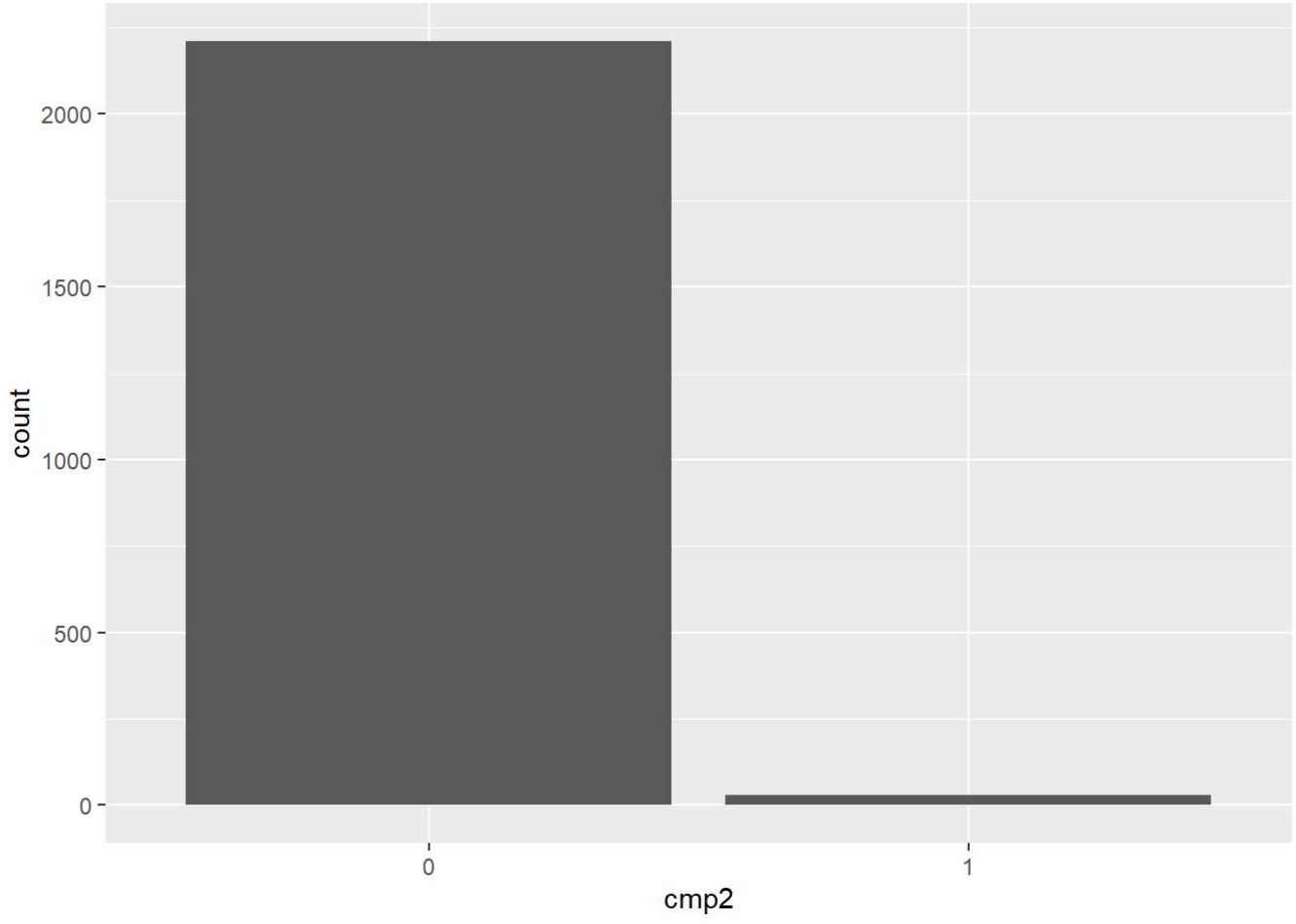
```
ggplot(retail,aes(cmp1))+geom_bar()
```



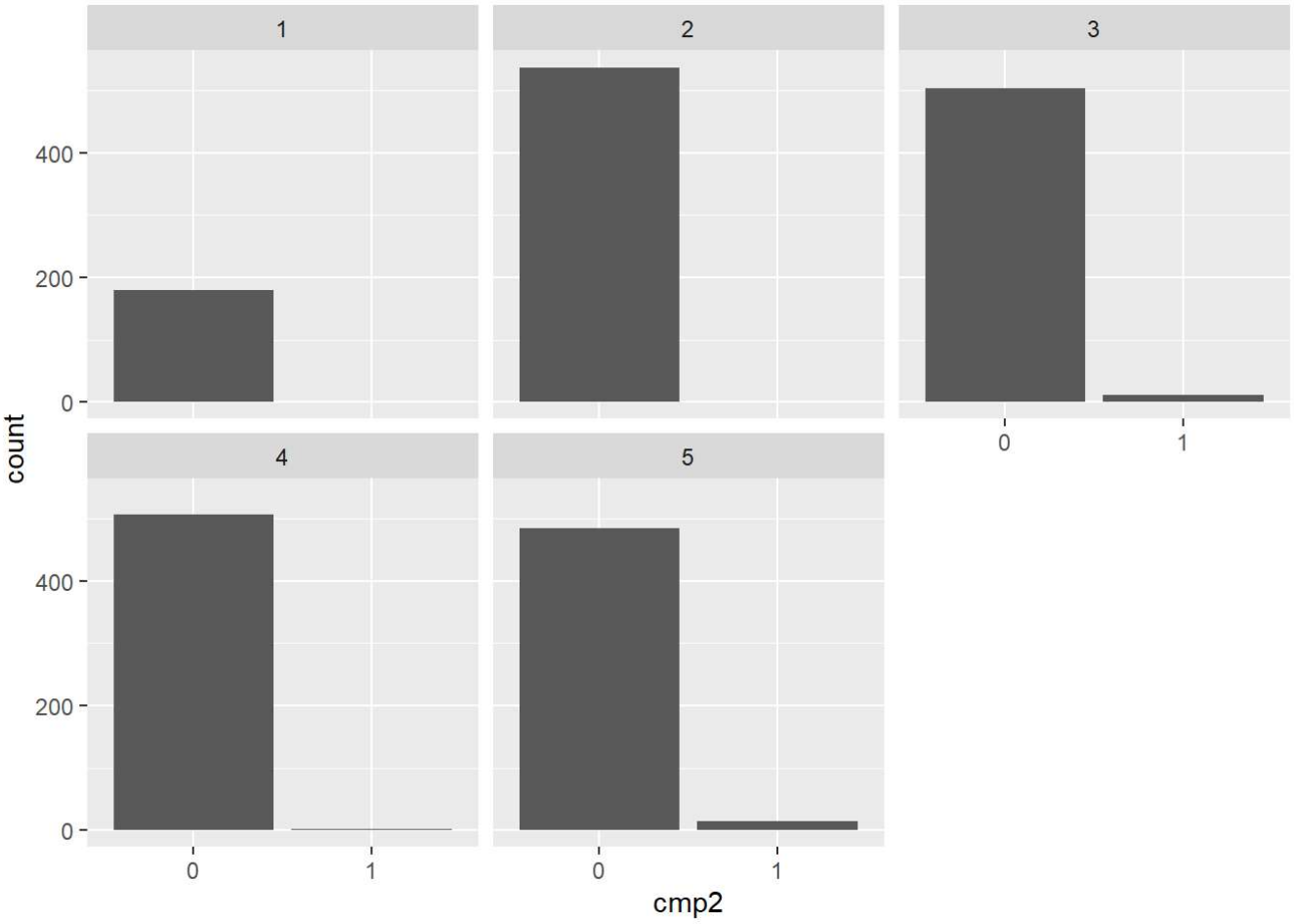

```
ggplot(retail,aes(cmp1))+geom_bar()+facet_wrap(~clusters5$cluster)
```



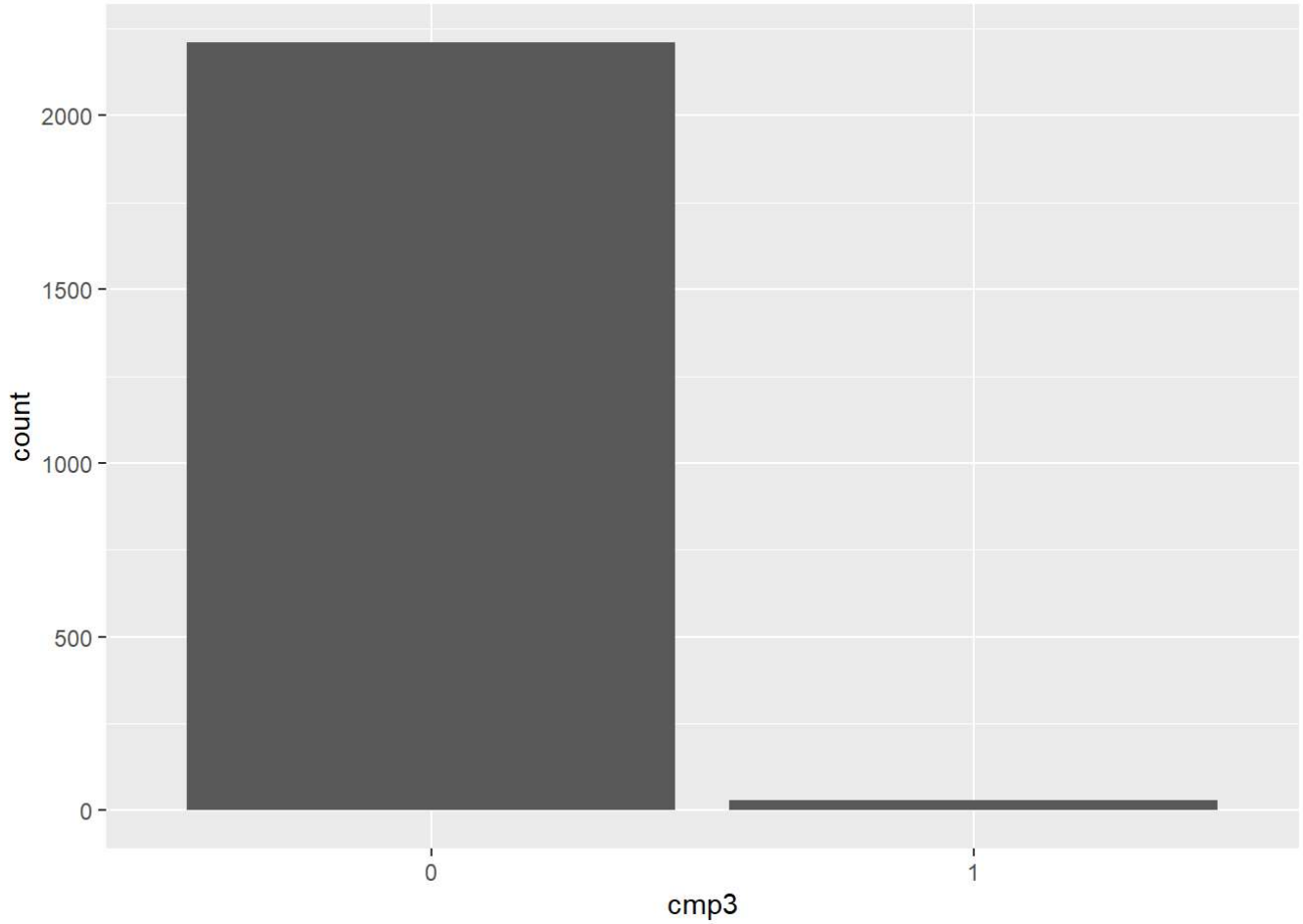

```
ggplot(retail,aes(cmp2))+geom_bar()
```



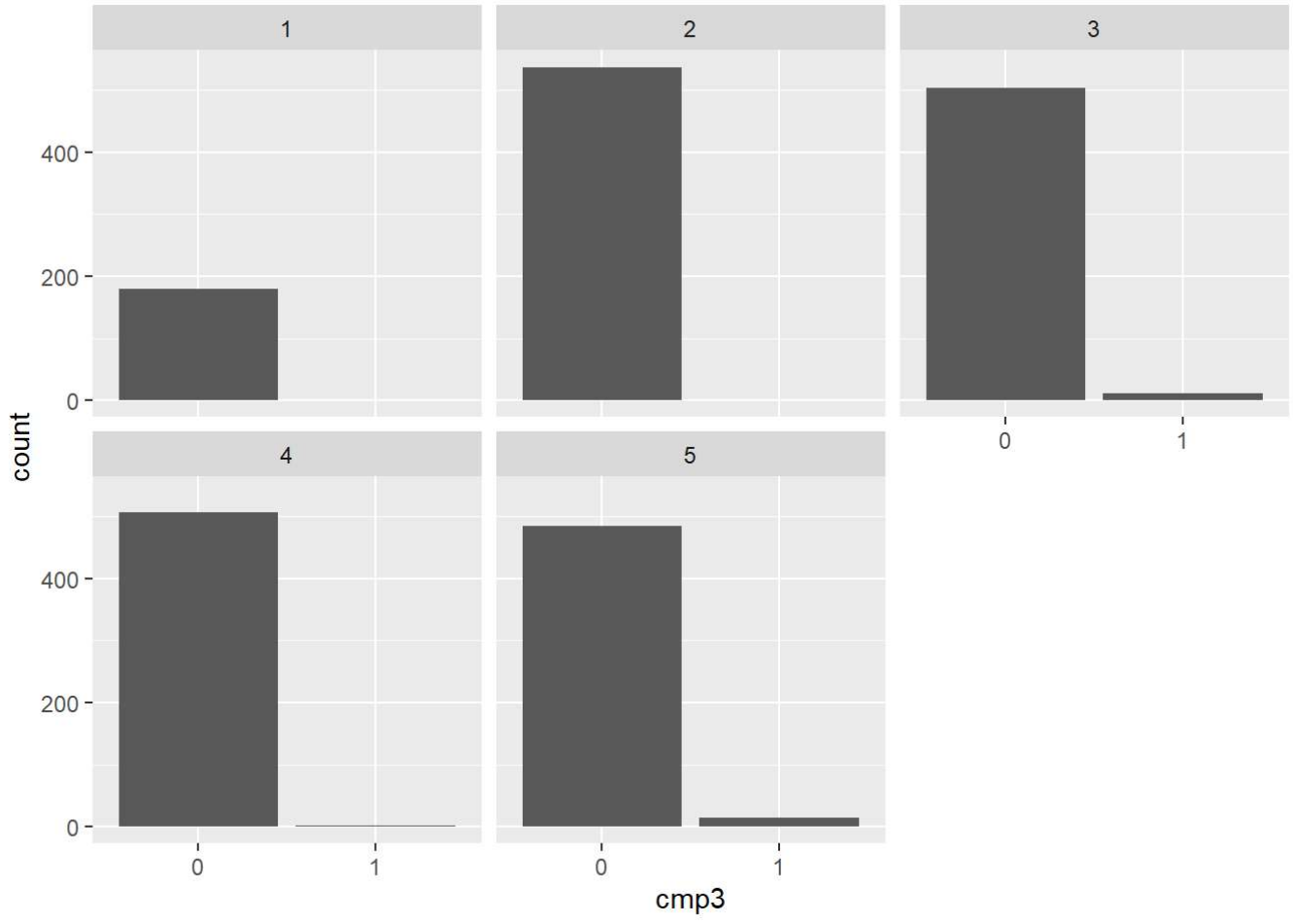
```
ggplot(retail,aes(cmp2))+geom_bar()+facet_wrap(~clusters5$cluster)
```



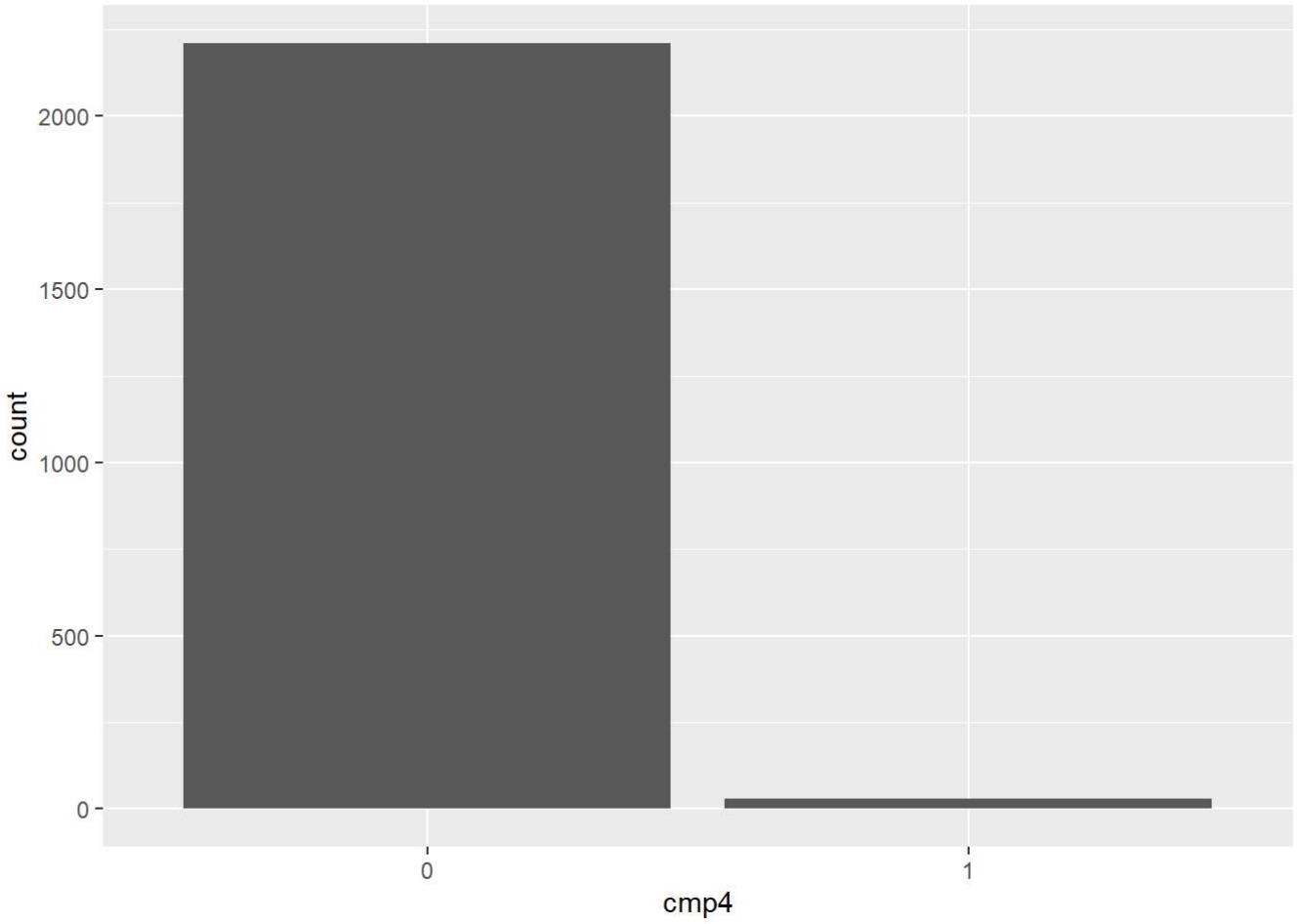

```
ggplot(retail,aes(cmp3))+geom_bar()
```



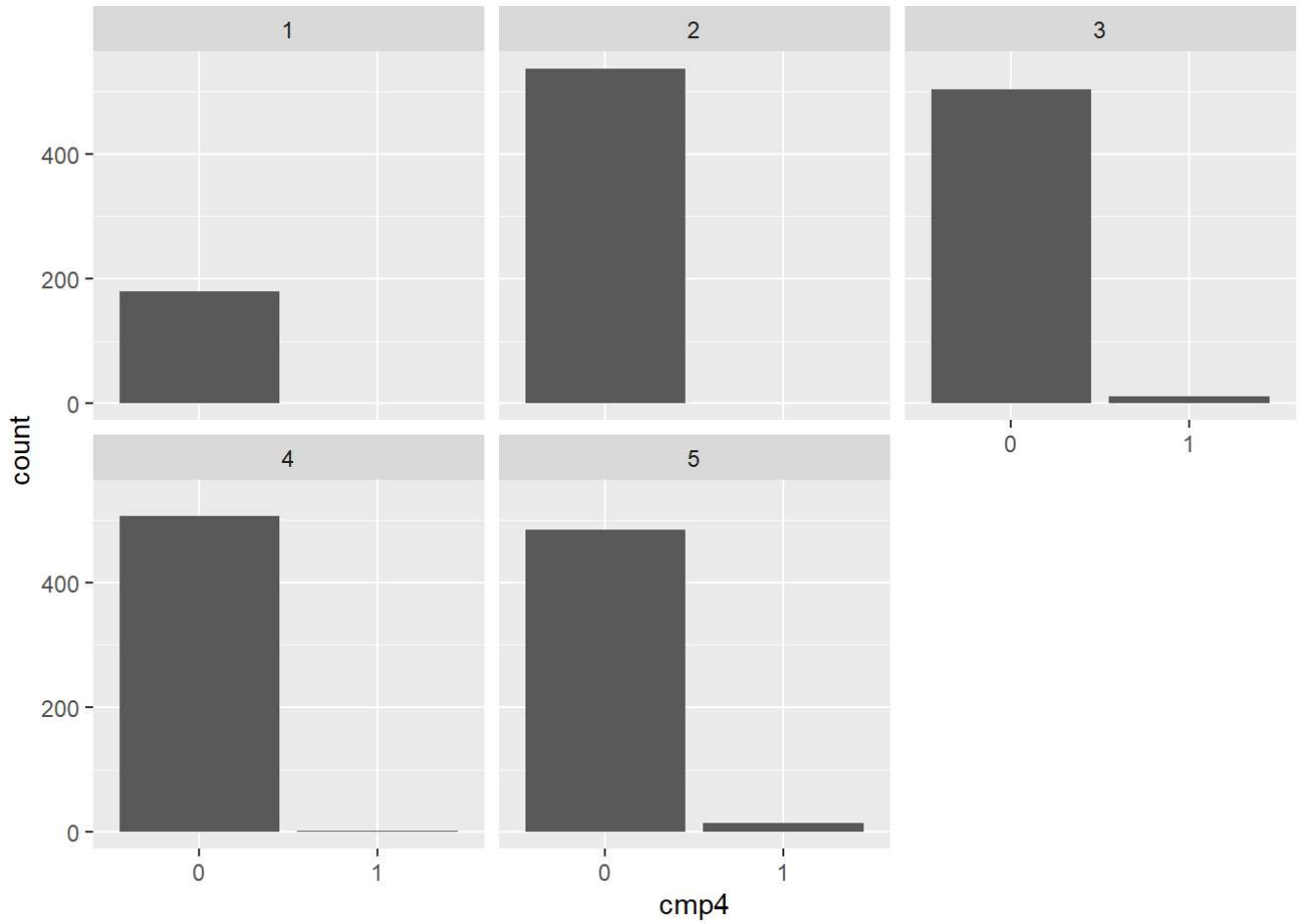

```
ggplot(retail,aes(cmp3))+geom_bar()+facet_wrap(~clusters5$cluster)
```



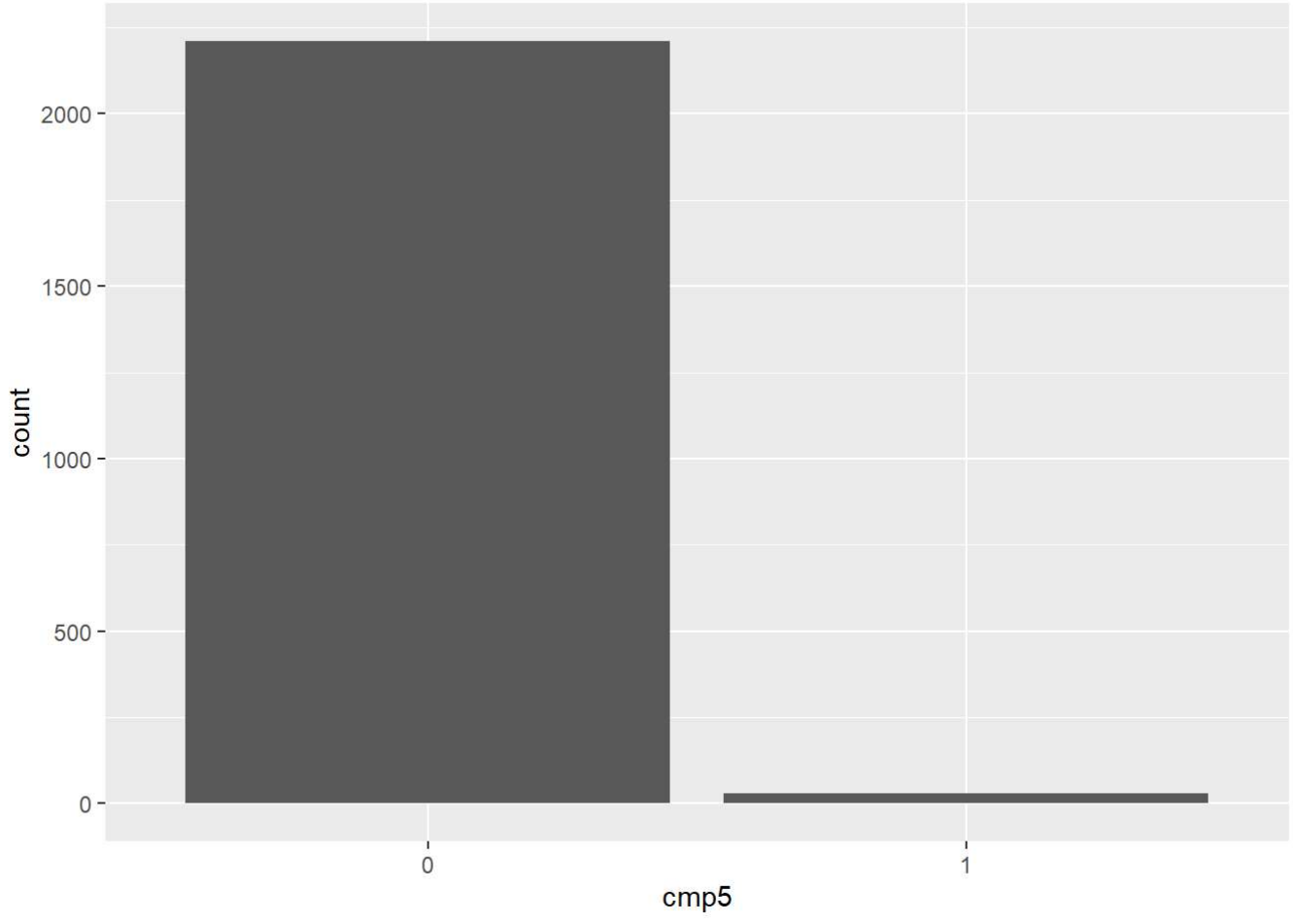
```
ggplot(retail,aes(cmp4))+geom_bar()
```



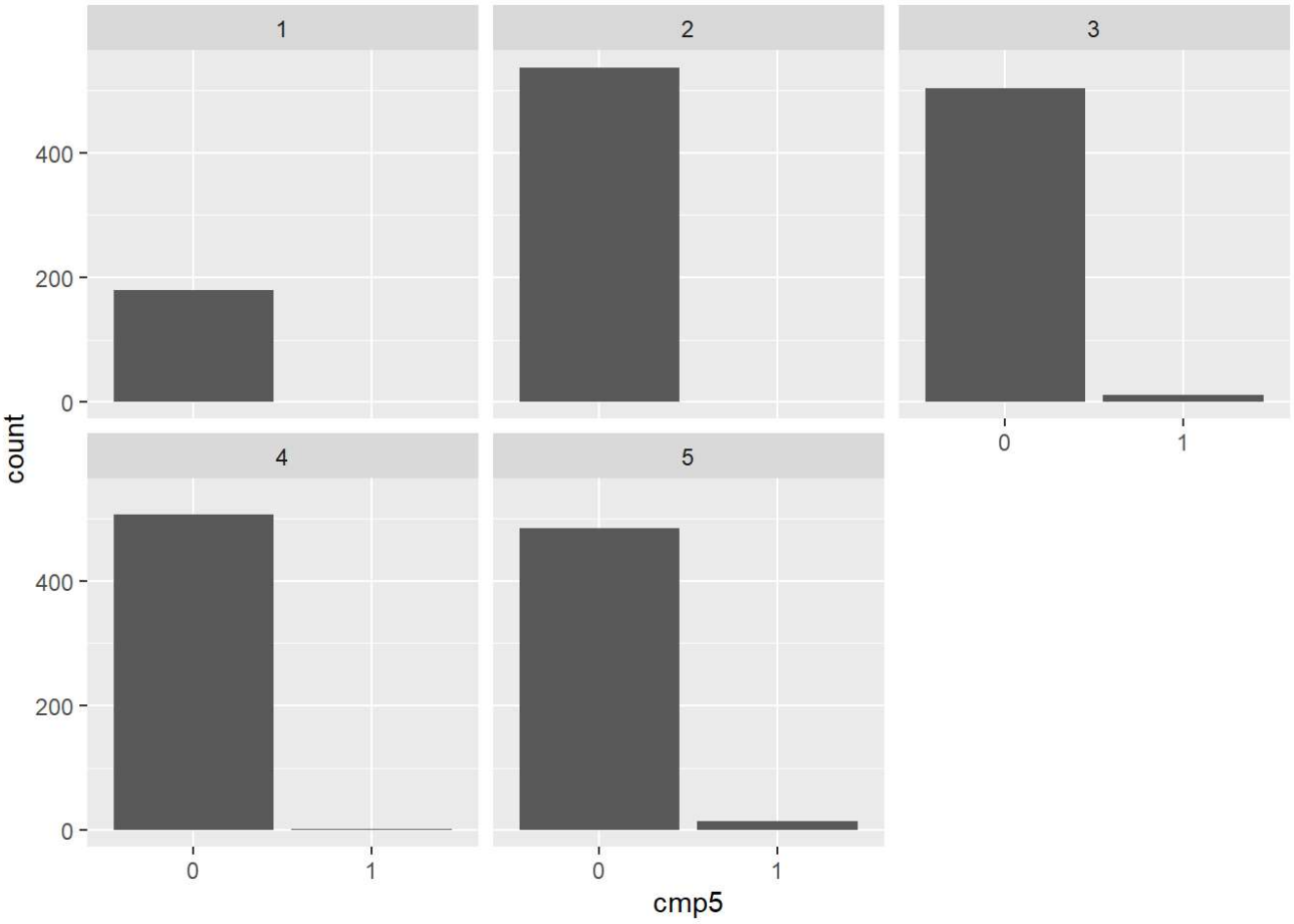

```
ggplot(retail,aes(cmp4))+geom_bar()+facet_wrap(~clusters5$cluster)
```



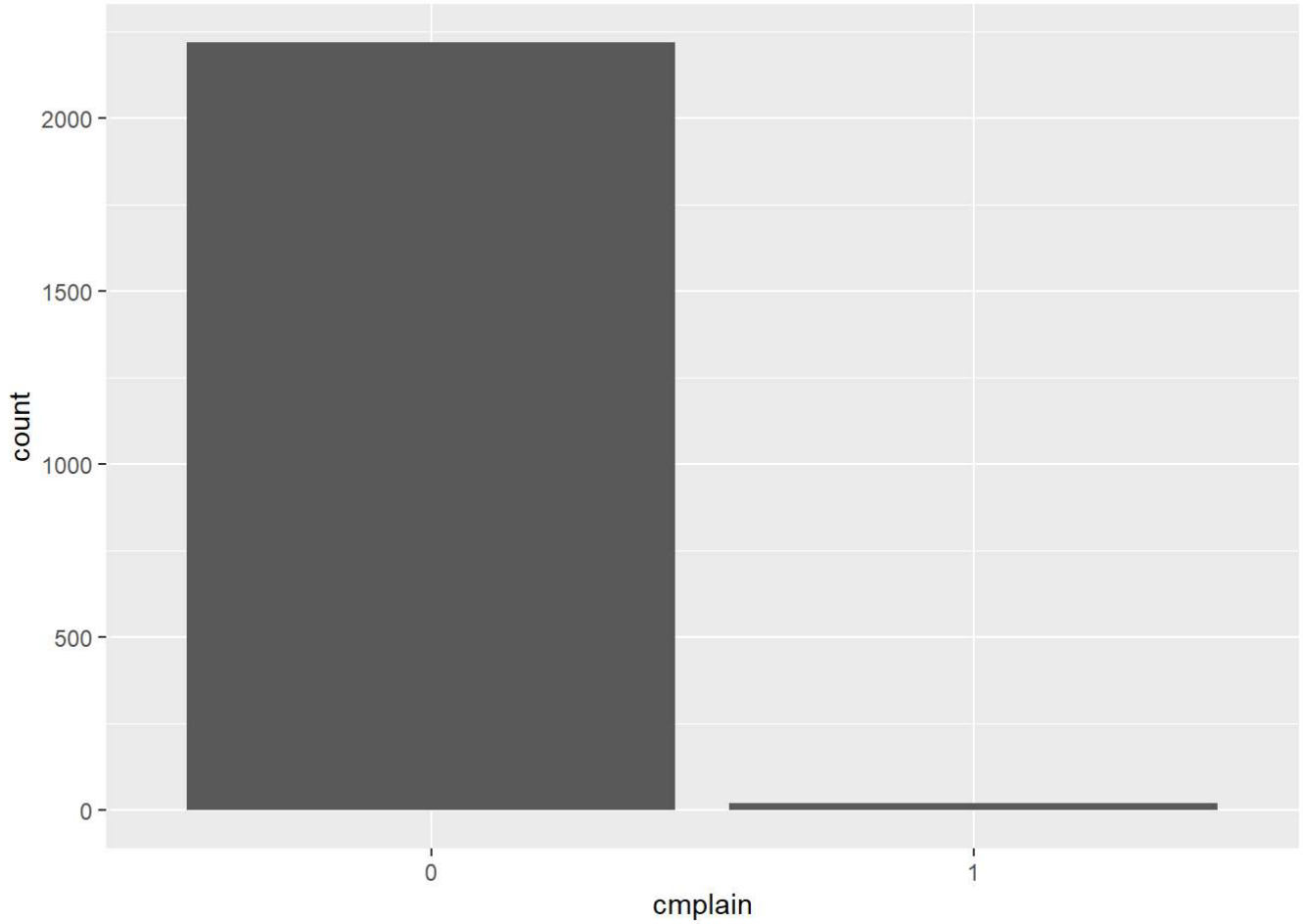

```
ggplot(retail,aes(cmp5))+geom_bar()
```



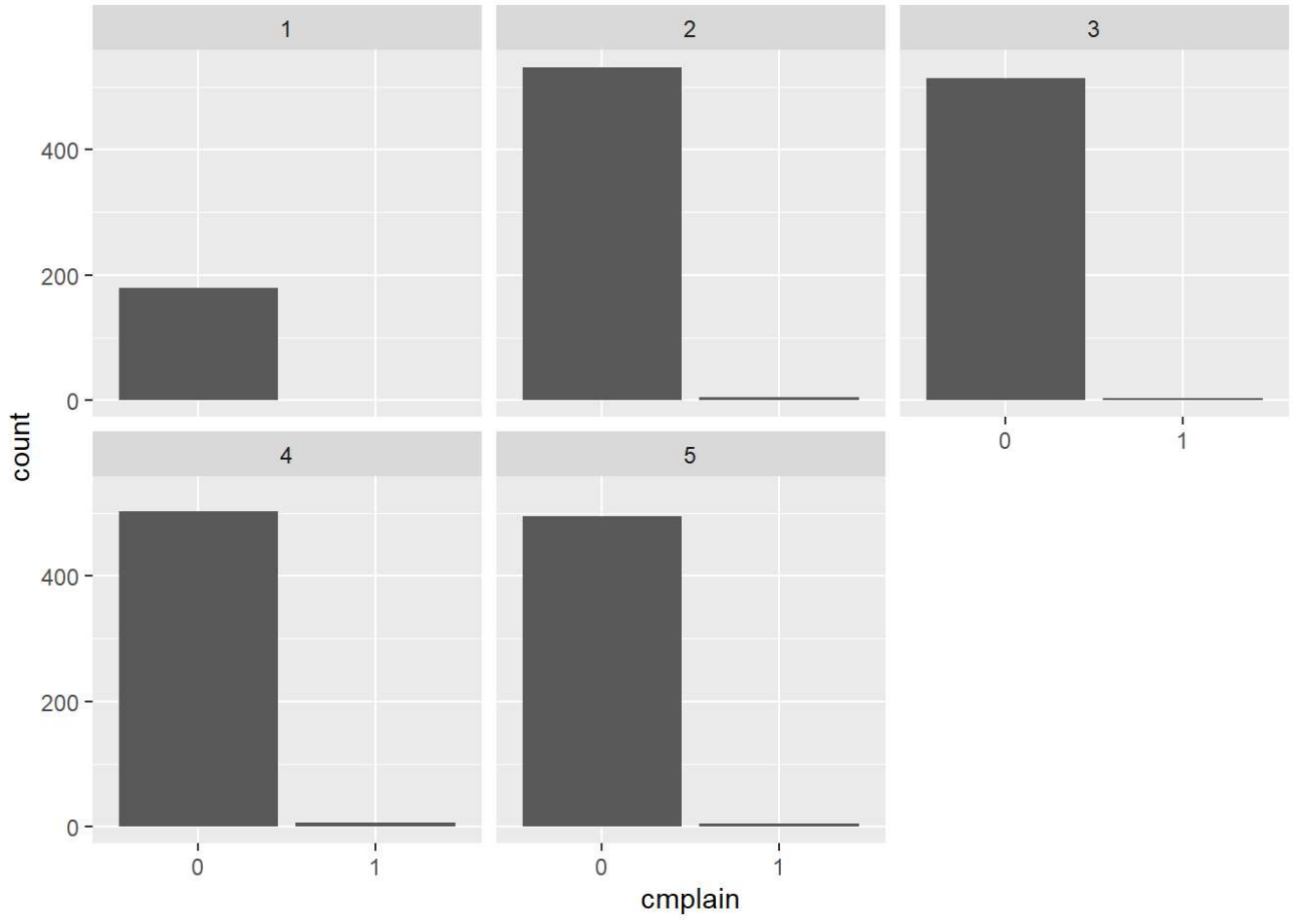
```
ggplot(retail,aes(cmp5))+geom_bar()+facet_wrap(~clusters5$cluster)
```




```
ggplot(retail,aes(cmplain))+geom_bar()
```




```
ggplot(retail,aes(cmplain))+geom_bar()+facet_wrap(~clusters5$cluster)
```



Data Transformation

```
data <- retail %>% mutate(age = 2022 - birth) %>% dplyr::select(-id, -birth, -dt_customer, -z_cost, -z_rev)
         %>% mutate_if(is.character, factor)


head(data)
```

education <fct>	mar_stat <fct>	income <dbl>	kids <dbl>	teens <dbl>	recency <dbl>	wines <dbl>	fruits <dbl>	meat <dbl>	fish <dbl>
Graduation	Single	58138	0	0	58	635	88	546	172
Graduation	Single	46344	1	1	38	11	1	6	2
Graduation	Partner	71613	0	0	26	426	49	127	111
Graduation	Partner	26646	1	0	26	11	4	20	10
PhD	Married	58293	1	0	94	173	43	118	46
Master	Partner	62513	0	1	16	520	42	98	0

6 rows | 1-10 of 25 columns

Partition my Data into 70/30 train/test split

```
set.seed(1234)

# -- performs our train / test split
split <- initial_split(data, prop = 0.7)

# -- extract the training data form our banana split
train <- training(split)
# -- extract the test data
test <- testing(split)

sprintf("Train PCT : %1.2f%%", nrow(train)/ nrow(data) * 100)
```

```
## [1] "Train PCT : 70.00%"
```

```
sprintf("Test  PCT : %1.2f%%", nrow(test)/ nrow(data) * 100)
```

```
## [1] "Test  PCT : 30.00%"
```

Recipe

```
# -- create our recipe --
data_recipe <- recipe(response ~ ., data = train) %>%
  step_impute_mode(all_nominal(), -all_outcomes()) %>%
  step_impute_median(all_numeric()) %>%
  step_dummy(all_nominal(), -all_outcomes()) %>%
  prep()

data_recipe
```

```
## Recipe
##
## Inputs:
##
##       role #variables
##    outcome          1
##  predictor         24
##
## Training data contained 1568 data points and no missing data.
##
## Operations:
##
## Mode imputation for education, mar_stat, cmp3, cmp4, cmp5, cmp1, cm... [trained]
## Median imputation for income, kids, teens, recency, wines, fruits, me... [trained]
## Dummy variables from education, mar_stat, cmp3, cmp4, cmp5, cmp1, cmp2, cmplain [trained]
```

Bake

```
# -- apply the recipe
bake_train <- bake(data_recipe, new_data = train)
bake_test  <- bake(data_recipe, new_data = test)
```

Fit Logistic Regression Model

```
## logistic code is here for reference and comparison
logistic_glm <-logistic_reg(mode = "classification") %>%
                  set_engine("glm") %>%
                  fit(response ~ ., data = train)

## check out your parameter estimates ...
tidy(logistic_glm) %>%
  mutate_at(c("estimate", "std.error", "statistic", "p.value"), round, 4)
```

term	estimate	std.error	statistic	p.value
<chr>	<dbl>	<dbl>	<dbl>	<dbl>
(Intercept)	-16.0478	882.7439	-0.0182	0.9855
educationBasic	-0.5283	0.8296	-0.6367	0.5243
educationGraduation	0.2780	0.3776	0.7362	0.4616
educationMaster	0.7381	0.4183	1.7644	0.0777
educationPhD	1.0662	0.4059	2.6267	0.0086
mar_statAlone	-0.1642	1248.3878	-0.0001	0.9999
mar_statDivorce	13.8662	882.7435	0.0157	0.9875
mar_statMarried	12.5689	882.7435	0.0142	0.9886
mar_statPartner	12.7141	882.7435	0.0144	0.9885
mar_statSingle	13.6755	882.7435	0.0155	0.9876

1-10 of 34 rows Previous 1 2 3 4 Next

Prep for Evaluation

```
# -- training
predict(logistic_glm, train, type = "prob") %>%
  bind_cols(.,predict(logistic_glm, train)) %>%
  bind_cols(.,train) -> scored_train_glm

head(scored_train_glm)
```

.pred_0	.pred_1	.pred_class	education	mar_stat	inco...	ki...	teens	recency	win...
<dbl>	<dbl>	<fct>	<fct>	<fct>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
0.8816814	0.118318647	0	Graduation	Partner	22434	1	0	25	4
0.8484840	0.151515961	0	Graduation	Widow	72298	0	0	52	625
0.9907476	0.009252399	0	Basic	Married	25443	1	0	82	1
0.9687851	0.031214930	0	2nCycle	Partner	14515	1	0	71	6
0.2456264	0.754373590	1	Graduation	Single	78499	0	0	12	912
0.8767287	0.123271313	0	PhD	Married	54549	0	1	8	216

6 rows | 1-10 of 28 columns

```
# -- testing
predict(logistic_glm, test, type = "prob") %>%
  bind_cols(.,predict(logistic_glm, test)) %>%
  bind_cols(.,test) -> scored_test_glm

head(scored_test_glm)
```

.pred_0 <dbl>	.pred_1 <dbl>	.pred_class <fct>	education <fct>	mar_stat <fct>	inco... <dbl>	kids <dbl>	teens <dbl>	recency <dbl>	win... <dbl>
0.3202707	0.67972927	1	Graduation	Single	58138	0	0	58	635
0.9478608	0.05213922	0	Graduation	Partner	71613	0	0	26	426
0.9383749	0.06162505	0	Graduation	Partner	26646	1	0	26	11
0.6779253	0.32207474	0	PhD	Partner	30351	1	0	19	14
0.6594218	0.34057822	0	PhD	Partner	5648	1	1	68	28
0.9834292	0.01657079	0	Basic	Married	7500	0	0	59	6

6 rows | 1-10 of 28 columns

Evaluate Logistic Regression Model

```
options(yardstick.event_first = FALSE)

# AUC: Train and Test
scored_train_glm %>%
  metrics(response, .pred_1, estimate = .pred_class) %>%
  mutate(part="training") %>%
  bind_rows(scored_test_glm %>%
              metrics(response, .pred_1, estimate = .pred_class) %>%
              mutate(part="testing")
  ) %>%
  filter(.metric %in% c("accuracy", "roc_auc"))
```

.metric <chr>	.estimator <chr>	.estimate <dbl>	part <chr>
accuracy	binary	0.8852041	training
roc_auc	binary	0.8691019	training
accuracy	binary	0.8735119	testing
roc_auc	binary	0.8113962	testing

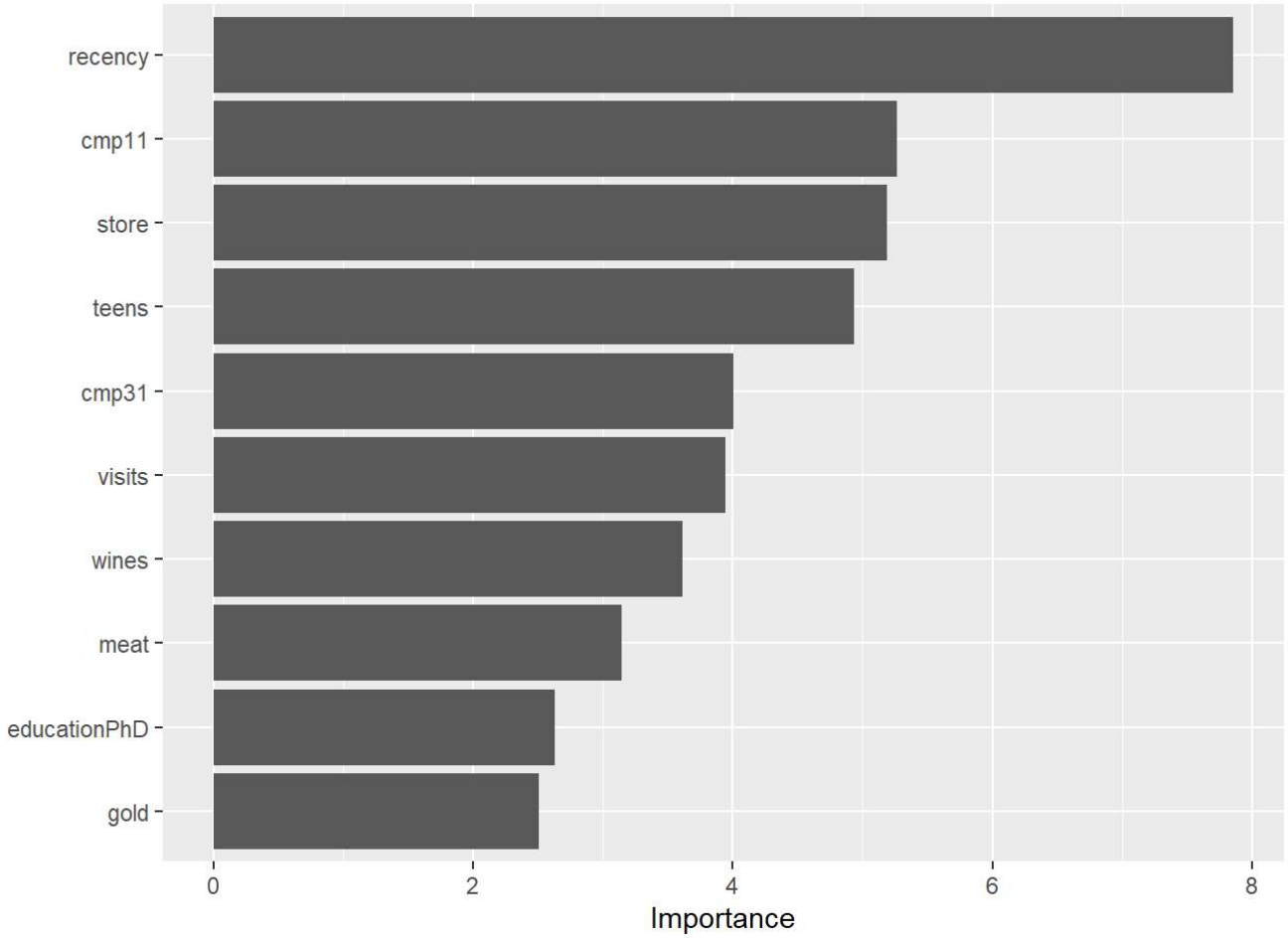
4 rows

```
# Variable Importance top 10 features
logistic_glm %>%
  vi()
```

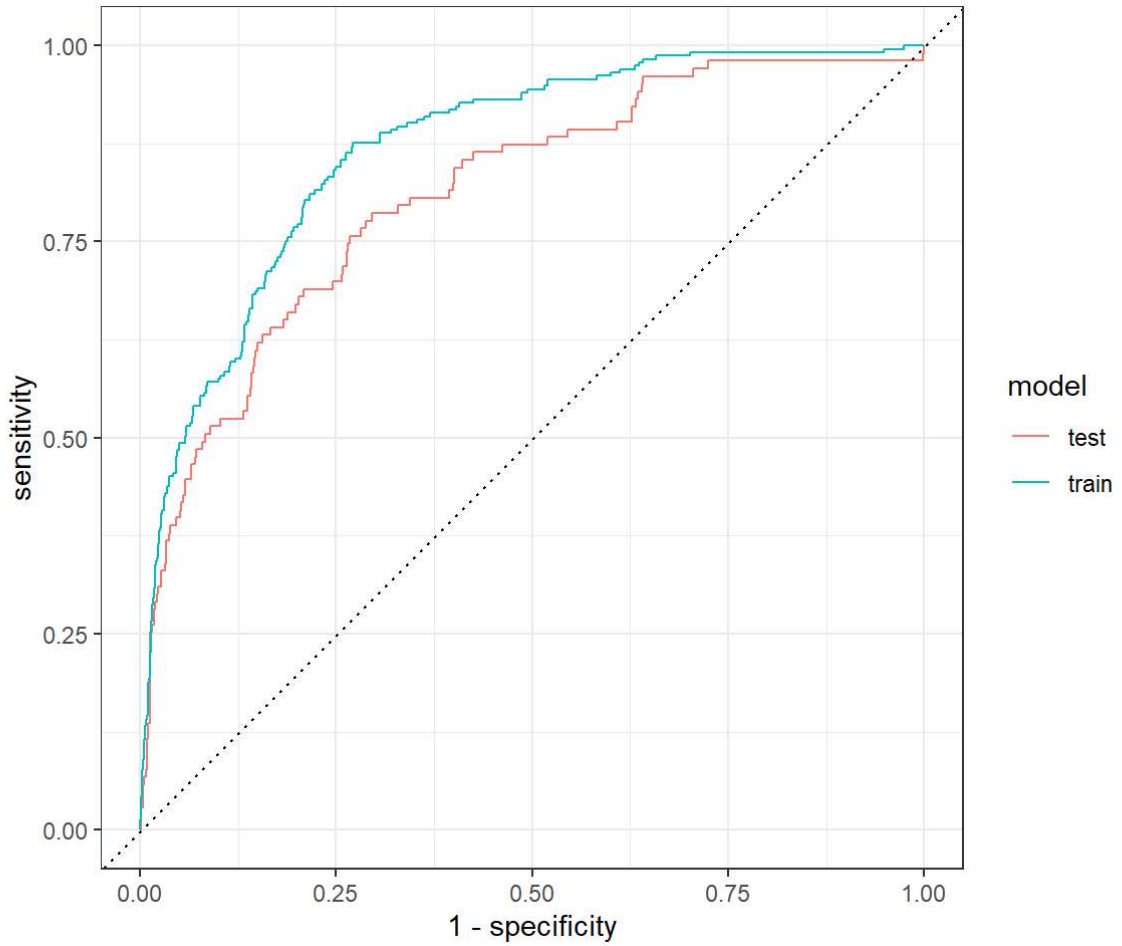
Variable	Importance	Sign
<chr>	<dbl>	<chr>
recency	7.8535544429	NEG
cmp11	5.2681227136	POS
store	5.1851947777	NEG
teens	4.9391418214	NEG
cmp31	4.0087637971	POS
visits	3.9457844825	POS
wines	3.6128027886	POS
meat	3.1422571937	POS
educationPhD	2.6267104631	POS
gold	2.5092635202	POS

1-10 of 30 rows Previous **1** 2 3 Next

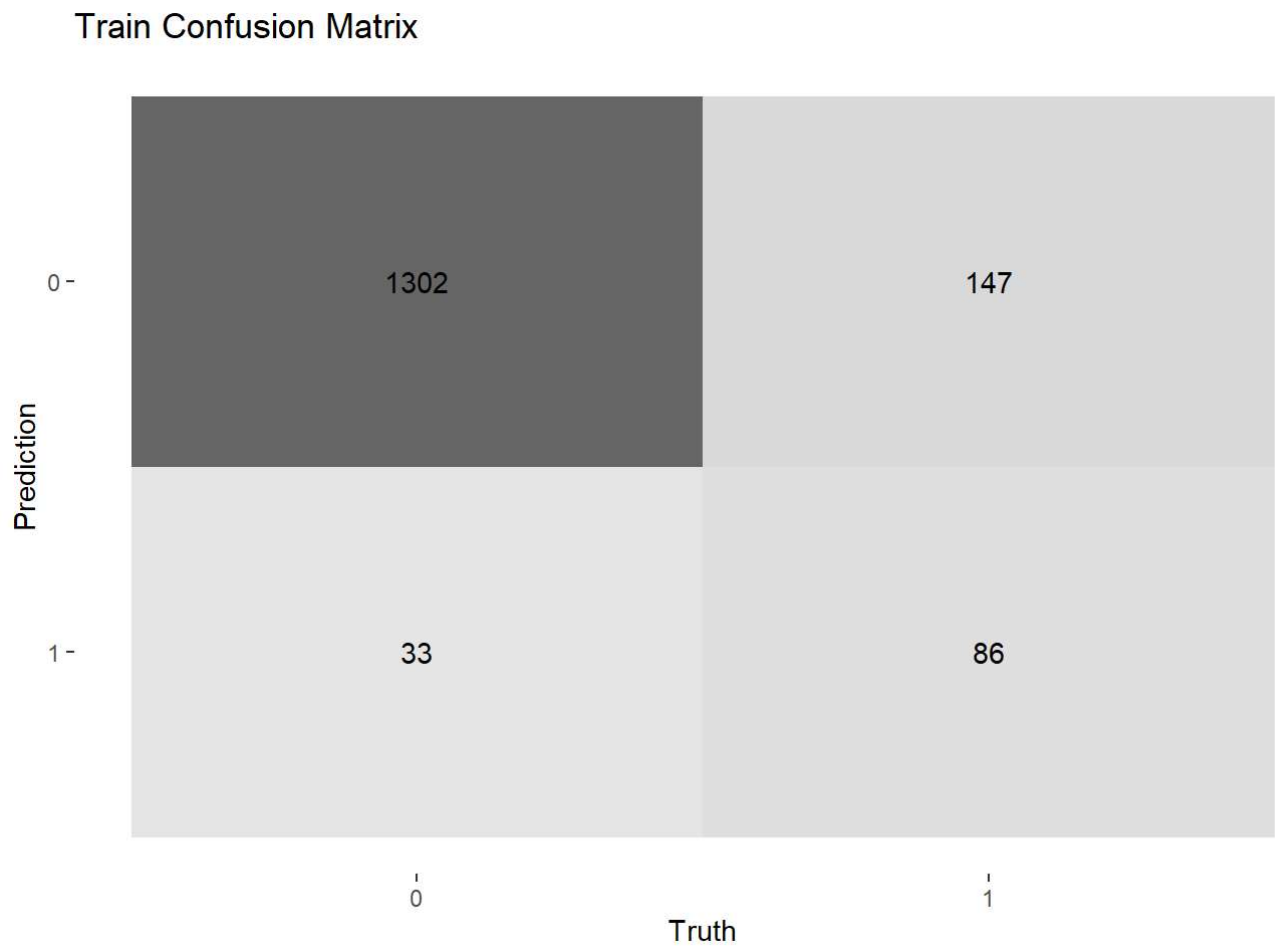
```
logistic_glm %>%
  vip(num_features = 10)
```



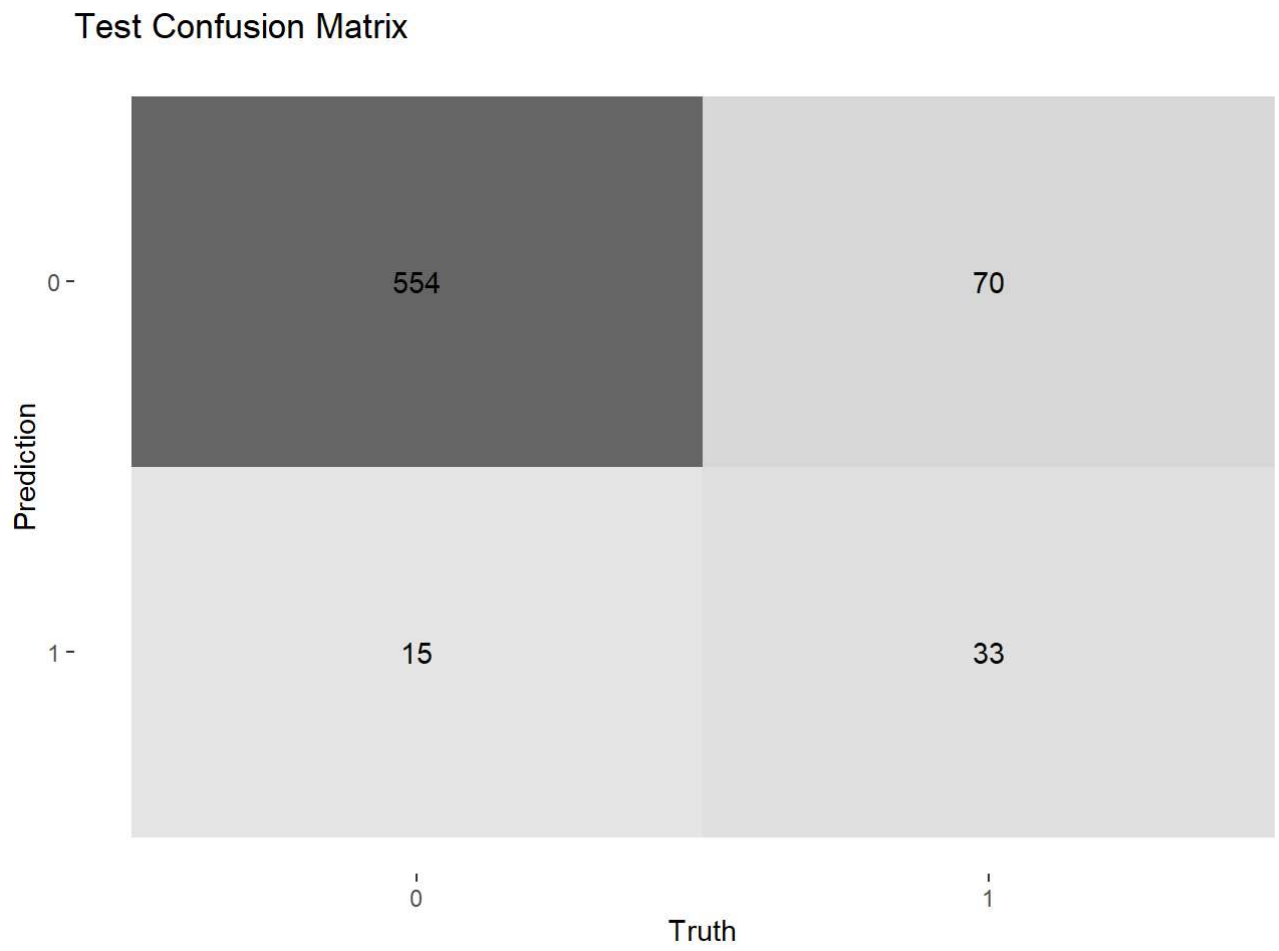

```
# ROC Charts
scored_train_glm %>%
  mutate(model = "train") %>%
  bind_rows(scored_test_glm %>%
              mutate(model="test")) %>%
  group_by(model) %>%
  roc_curve(response, .pred_1) %>%
  autoplot()
```




```
# Confusion Matrices
scored_train_glm %>%
  conf_mat(response, .pred_class) %>%
  autoplot(type = "heatmap") +
  labs(title="Train Confusion Matrix")
```




```
scored_test_glm %>%
  conf_mat(response, .pred_class) %>%
  autoplot(type = "heatmap") +
  labs(title="Test Confusion Matrix")
```



Reduced Model - backward elimination (stepAIC)

```
## Use stepwise selection to reduce the model

steplog <- glm(response ~ ., data = bake_train, family=binomial(link="logit"))
step <- stepAIC(steplog, direction="both")
```

```
## Start:  AIC=978.86
## response ~ income + kids + teens + recency + wines + fruits +
##     meat + fish + sweets + gold + deals + web + catalog + store +
##     visits + age + education_Basic + education_Graduation + education_Master +
##     education_PhD + mar_stat_Alone + mar_stat_Divorce + mar_stat_Married +
##     mar_stat_Partner + mar_stat_Single + mar_stat_Widow + mar_stat_YOLO +
##     cmp3_X1 + cmp4_X1 + cmp5_X1 + cmp1_X1 + cmp2_X1 + cmplain_X1
##
##
## Step:  AIC=978.86
## response ~ income + kids + teens + recency + wines + fruits +
##     meat + fish + sweets + gold + deals + web + catalog + store +
##     visits + age + education_Basic + education_Graduation + education_Master +
##     education_PhD + mar_stat_Alone + mar_stat_Divorce + mar_stat_Married +
##     mar_stat_Partner + mar_stat_Single + mar_stat_Widow + mar_stat_YOLO +
##     cmp3_X1 + cmp4_X1 + cmp5_X1 + cmp1_X1 + cmplain_X1
##
##
## Step:  AIC=978.86
## response ~ income + kids + teens + recency + wines + fruits +
##     meat + fish + sweets + gold + deals + web + catalog + store +
##     visits + age + education_Basic + education_Graduation + education_Master +
##     education_PhD + mar_stat_Alone + mar_stat_Divorce + mar_stat_Married +
##     mar_stat_Partner + mar_stat_Single + mar_stat_Widow + mar_stat_YOLO +
##     cmp3_X1 + cmp4_X1 + cmp1_X1 + cmplain_X1
##
##
## Step:  AIC=978.86
## response ~ income + kids + teens + recency + wines + fruits +
##     meat + fish + sweets + gold + deals + web + catalog + store +
##     visits + age + education_Basic + education_Graduation + education_Master +
##     education_PhD + mar_stat_Alone + mar_stat_Divorce + mar_stat_Married +
##     mar_stat_Partner + mar_stat_Single + mar_stat_Widow + mar_stat_YOLO +
##     cmp3_X1 + cmp1_X1 + cmplain_X1
##
##                        Df Deviance     AIC
## - mar_stat_Alone        1   916.86  976.86
## - kids                  1   916.87  976.87
## - sweets                1   916.98  976.98
## - age                   1   917.08  977.08
## - mar_stat_Married      1   917.11  977.11
## - mar_stat_Partner      1   917.15  977.15
## - cmplain_X1            1   917.29  977.29
## - education_Basic       1   917.30  977.30
## - education_Graduation  1   917.43  977.43
## - income                1   917.48  977.48
```

```
## - mar_stat_Single       1   917.53  977.53
## - catalog               1   917.58  977.58
## - mar_stat_Divorce      1   917.64  977.64
## - fish                  1   917.78  977.78
## - mar_stat_Widow        1   917.97  977.97
## - mar_stat_YOLO         1   918.82  978.82
## <none>                      916.86  978.86
## - fruits                1   919.67  979.67
## - education_Master      1   920.18  980.18
## - web                   1   920.60  980.60
## - deals                 1   921.15  981.15
## - gold                  1   923.07  983.07
## - education_PhD         1   924.56  984.56
## - meat                  1   927.03  987.03
## - wines                 1   929.88  989.88
## - visits                1   932.70  992.70
## - cmp3_X1               1   934.63  994.63
## - teens                 1   943.68 1003.68
## - cmp1_X1               1   944.81 1004.81
## - store                 1   946.42 1006.42
## - recency               1   986.28 1046.28
##
## Step:  AIC=976.86
## response ~ income + kids + teens + recency + wines + fruits +
##     meat + fish + sweets + gold + deals + web + catalog + store +
##     visits + age + education_Basic + education_Graduation + education_Master +
##     education_PhD + mar_stat_Divorce + mar_stat_Married + mar_stat_Partner +
##     mar_stat_Single + mar_stat_Widow + mar_stat_YOLO + cmp3_X1 +
##     cmp1_X1 + cmplain_X1
##
##                        Df Deviance     AIC
## - kids                  1   916.87  974.87
## - sweets                1   916.98  974.98
## - age                   1   917.08  975.08
## - cmplain_X1            1   917.29  975.29
## - education_Basic       1   917.30  975.30
## - mar_stat_Married      1   917.41  975.41
## - education_Graduation  1   917.43  975.43
## - income                1   917.48  975.48
## - mar_stat_Partner      1   917.48  975.48
## - catalog               1   917.58  975.58
## - fish                  1   917.78  975.78
## - mar_stat_Single       1   918.30  976.30
## - mar_stat_Divorce      1   918.53  976.53
## <none>                      916.86  976.86
## - mar_stat_Widow        1   919.19  977.19
## - fruits                1   919.67  977.67
```

```
## - mar_stat_YOLO         1   919.69  977.69
## - education_Master      1   920.18  978.18
## - web                   1   920.60  978.60
## + mar_stat_Alone        1   916.86  978.86
## - deals                 1   921.15  979.15
## - gold                  1   923.07  981.07
## - education_PhD         1   924.56  982.56
## - meat                  1   927.03  985.03
## - wines                 1   929.88  987.88
## - visits                1   932.70  990.70
## - cmp3_X1               1   934.63  992.63
## - teens                 1   943.68 1001.68
## - cmp1_X1               1   944.81 1002.81
## - store                 1   946.42 1004.42
## - recency               1   986.28 1044.28
##
## Step:  AIC=974.87
## response ~ income + teens + recency + wines + fruits + meat +
##     fish + sweets + gold + deals + web + catalog + store + visits +
##     age + education_Basic + education_Graduation + education_Master +
##     education_PhD + mar_stat_Divorce + mar_stat_Married + mar_stat_Partner +
##     mar_stat_Single + mar_stat_Widow + mar_stat_YOLO + cmp3_X1 +
##     cmp1_X1 + cmplain_X1
##
##                        Df Deviance     AIC
## - sweets                1   916.99  972.99
## - age                   1   917.10  973.10
## - cmplain_X1            1   917.30  973.30
## - education_Basic       1   917.31  973.31
## - mar_stat_Married      1   917.41  973.41
## - education_Graduation  1   917.44  973.44
## - mar_stat_Partner      1   917.49  973.49
## - income                1   917.51  973.51
## - catalog               1   917.59  973.59
## - fish                  1   917.79  973.79
## - mar_stat_Single       1   918.30  974.30
## - mar_stat_Divorce      1   918.53  974.53
## <none>                      916.87  974.87
## - mar_stat_Widow        1   919.19  975.19
## - fruits                1   919.67  975.67
## - mar_stat_YOLO         1   919.69  975.69
## - education_Master      1   920.22  976.22
## - web                   1   920.65  976.65
## + kids                  1   916.86  976.86
## + mar_stat_Alone        1   916.87  976.87
## - deals                 1   922.03  978.03
## - gold                  1   923.08  979.08
```

```
## - education_PhD         1   924.59  980.59
## - meat                  1   927.12  983.12
## - wines                 1   930.08  986.08
## - visits                1   933.03  989.03
## - cmp3_X1               1   934.66  990.66
## - cmp1_X1               1   945.01 1001.01
## - teens                 1   945.21 1001.21
## - store                 1   947.42 1003.42
## - recency               1   986.35 1042.35
##
## Step:  AIC=972.99
## response ~ income + teens + recency + wines + fruits + meat +
##     fish + gold + deals + web + catalog + store + visits + age +
##     education_Basic + education_Graduation + education_Master +
##     education_PhD + mar_stat_Divorce + mar_stat_Married + mar_stat_Partner +
##     mar_stat_Single + mar_stat_Widow + mar_stat_YOLO + cmp3_X1 +
##     cmp1_X1 + cmplain_X1
##
##                        Df Deviance     AIC
## - age                   1   917.21  971.21
## - cmplain_X1            1   917.42  971.42
## - education_Basic       1   917.44  971.44
## - mar_stat_Married      1   917.52  971.52
## - education_Graduation  1   917.56  971.56
## - income                1   917.59  971.59
## - mar_stat_Partner      1   917.59  971.59
## - catalog               1   917.73  971.73
## - fish                  1   918.13  972.13
## - mar_stat_Single       1   918.39  972.39
## - mar_stat_Divorce      1   918.61  972.61
## <none>                      916.99  972.99
## - mar_stat_Widow        1   919.26  973.26
## - fruits                1   919.68  973.68
## - mar_stat_YOLO         1   919.78  973.78
## - education_Master      1   920.41  974.41
## - web                   1   920.68  974.68
## + sweets                1   916.87  974.87
## + kids                  1   916.98  974.98
## + mar_stat_Alone        1   916.99  974.99
## - deals                 1   922.34  976.34
## - gold                  1   923.17  977.17
## - education_PhD         1   924.80  978.80
## - meat                  1   927.14  981.14
## - wines                 1   930.41  984.41
## - visits                1   933.39  987.39
## - cmp3_X1               1   935.01  989.01
## - cmp1_X1               1   945.01  999.01
```

```
## - teens                1   945.22  999.22
## - store                1   947.51 1001.51
## - recency              1   986.98 1040.98
##
## Step:  AIC=971.21
## response ~ income + teens + recency + wines + fruits + meat +
##     fish + gold + deals + web + catalog + store + visits + education_Basic +
##     education_Graduation + education_Master + education_PhD +
##     mar_stat_Divorce + mar_stat_Married + mar_stat_Partner +
##     mar_stat_Single + mar_stat_Widow + mar_stat_YOLO + cmp3_X1 +
##     cmp1_X1 + cmplain_X1
##
##                        Df Deviance     AIC
## - education_Basic       1   917.64  969.64
## - cmplain_X1            1   917.67  969.67
## - education_Graduation  1   917.74  969.74
## - mar_stat_Married      1   917.75  969.75
## - income                1   917.79  969.79
## - mar_stat_Partner      1   917.82  969.82
## - catalog               1   917.91  969.91
## - fish                  1   918.43  970.43
## - mar_stat_Single       1   918.65  970.65
## - mar_stat_Divorce      1   918.87  970.87
## <none>                      917.21  971.21
## - mar_stat_Widow        1   919.48  971.48
## - fruits                1   919.89  971.89
## - mar_stat_YOLO         1   920.07  972.07
## - education_Master      1   920.50  972.50
## - web                   1   920.75  972.75
## + age                   1   916.99  972.99
## + sweets                1   917.10  973.10
## + kids                  1   917.19  973.19
## + mar_stat_Alone        1   917.21  973.21
## - deals                 1   922.69  974.69
## - gold                  1   923.44  975.44
## - education_PhD         1   924.81  976.81
## - meat                  1   927.48  979.48
## - wines                 1   930.57  982.57
## - visits                1   933.82  985.82
## - cmp3_X1               1   935.28  987.28
## - cmp1_X1               1   945.62  997.62
## - store                 1   947.62  999.62
## - teens                 1   948.42 1000.42
## - recency               1   987.01 1039.01
##
## Step:  AIC=969.64
## response ~ income + teens + recency + wines + fruits + meat +
```

```
##     fish + gold + deals + web + catalog + store + visits + education_Graduation +
##     education_Master + education_PhD + mar_stat_Divorce + mar_stat_Married +
##     mar_stat_Partner + mar_stat_Single + mar_stat_Widow + mar_stat_YOLO +
##     cmp3_X1 + cmp1_X1 + cmplain_X1
##
##                        Df Deviance     AIC
## - cmplain_X1            1   918.15  968.15
## - mar_stat_Married      1   918.19  968.19
## - mar_stat_Partner      1   918.26  968.26
## - catalog               1   918.34  968.34
## - income                1   918.35  968.35
## - fish                  1   918.81  968.81
## - education_Graduation  1   918.93  968.93
## - mar_stat_Single       1   919.08  969.08
## - mar_stat_Divorce      1   919.32  969.32
## <none>                      917.64  969.64
## - mar_stat_Widow        1   919.92  969.92
## - fruits                1   920.30  970.30
## - mar_stat_YOLO         1   920.51  970.51
## + education_Basic       1   917.21  971.21
## - web                   1   921.23  971.23
## + age                   1   917.44  971.44
## + sweets                1   917.53  971.53
## + kids                  1   917.62  971.62
## + mar_stat_Alone        1   917.64  971.64
## - education_Master      1   922.69  972.69
## - deals                 1   923.15  973.15
## - gold                  1   923.90  973.90
## - meat                  1   927.82  977.82
## - education_PhD         1   928.39  978.39
## - wines                 1   930.77  980.77
## - visits                1   934.43  984.43
## - cmp3_X1               1   935.85  985.85
## - cmp1_X1               1   946.08  996.08
## - store                 1   947.82  997.82
## - teens                 1   948.63  998.63
## - recency               1   987.53 1037.53
##
## Step:  AIC=968.15
## response ~ income + teens + recency + wines + fruits + meat +
##     fish + gold + deals + web + catalog + store + visits + education_Graduation +
##     education_Master + education_PhD + mar_stat_Divorce + mar_stat_Married +
##     mar_stat_Partner + mar_stat_Single + mar_stat_Widow + mar_stat_YOLO +
##     cmp3_X1 + cmp1_X1
##
##                        Df Deviance     AIC
## - mar_stat_Married      1   918.70  966.70
```

```
## - mar_stat_Partner      1   918.77  966.77
## - catalog               1   918.84  966.84
## - income                1   918.85  966.85
## - fish                  1   919.33  967.33
## - education_Graduation  1   919.35  967.35
## - mar_stat_Single       1   919.60  967.60
## - mar_stat_Divorce      1   919.83  967.83
## <none>                      918.15  968.15
## - mar_stat_Widow        1   920.43  968.43
## - fruits                1   920.82  968.82
## - mar_stat_YOLO         1   921.02  969.02
## + cmplain_X1            1   917.64  969.64
## + education_Basic       1   917.67  969.67
## - web                   1   921.72  969.72
## + age                   1   917.93  969.93
## + sweets                1   918.04  970.04
## + kids                  1   918.13  970.13
## + mar_stat_Alone        1   918.15  970.15
## - education_Master      1   923.04  971.04
## - deals                 1   923.70  971.70
## - gold                  1   924.44  972.44
## - meat                  1   928.33  976.33
## - education_PhD         1   928.79  976.79
## - wines                 1   931.21  979.21
## - visits                1   934.92  982.92
## - cmp3_X1               1   936.36  984.36
## - cmp1_X1               1   946.58  994.58
## - store                 1   948.41  996.41
## - teens                 1   949.33  997.33
## - recency               1   988.22 1036.22
##
## Step:  AIC=966.7
## response ~ income + teens + recency + wines + fruits + meat +
##     fish + gold + deals + web + catalog + store + visits + education_Graduation +
##     education_Master + education_PhD + mar_stat_Divorce + mar_stat_Partner +
##     mar_stat_Single + mar_stat_Widow + mar_stat_YOLO + cmp3_X1 +
##     cmp1_X1
##
##                         Df Deviance     AIC
## - mar_stat_Partner      1   919.05  965.05
## - income                1   919.39  965.39
## - catalog               1   919.41  965.41
## - education_Graduation  1   919.90  965.90
## - fish                  1   919.95  965.95
## <none>                      918.70  966.70
## - fruits                1   921.40  967.40
## - mar_stat_YOLO         1   921.52  967.52
```

```
## + mar_stat_Married      1   918.15  968.15
## + cmplain_X1            1   918.19  968.19
## + education_Basic       1   918.21  968.21
## - web                   1   922.21  968.21
## + mar_stat_Alone        1   918.41  968.41
## + age                   1   918.46  968.46
## + sweets                1   918.60  968.60
## + kids                  1   918.68  968.68
## - education_Master      1   923.46  969.46
## - deals                 1   924.18  970.18
## - gold                  1   924.95  970.95
## - meat                  1   928.87  974.87
## - education_PhD         1   929.33  975.33
## - wines                 1   931.84  977.84
## - visits                1   935.51  981.51
## - mar_stat_Widow        1   935.79  981.79
## - cmp3_X1               1   936.87  982.87
## - mar_stat_Divorce      1   938.71  984.71
## - mar_stat_Single       1   942.90  988.90
## - cmp1_X1               1   947.23  993.23
## - store                 1   948.84  994.84
## - teens                 1   949.86  995.86
## - recency               1   988.90 1034.90
##
## Step:  AIC=965.05
## response ~ income + teens + recency + wines + fruits + meat +
##     fish + gold + deals + web + catalog + store + visits + education_Graduation +
##     education_Master + education_PhD + mar_stat_Divorce + mar_stat_Single +
##     mar_stat_Widow + mar_stat_YOLO + cmp3_X1 + cmp1_X1
##
##                         Df Deviance     AIC
## - income                 1   919.71  963.71
## - catalog                1   919.75  963.75
## - fish                   1   920.24  964.24
## - education_Graduation   1   920.28  964.28
## <none>                       919.05  965.05
## - fruits                 1   921.74  965.74
## - mar_stat_YOLO          1   921.79  965.79
## + cmplain_X1             1   918.54  966.54
## + education_Basic        1   918.55  966.55
## + mar_stat_Partner       1   918.70  966.70
## - web                    1   922.71  966.71
## + mar_stat_Alone         1   918.73  966.73
## + mar_stat_Married       1   918.77  966.77
## + age                    1   918.84  966.84
## + sweets                 1   918.96  966.96
## + kids                   1   919.03  967.03
```

```
## - education_Master      1   923.89  967.89
## - deals                 1   924.39  968.39
## - gold                  1   925.28  969.28
## - meat                  1   929.25  973.25
## - education_PhD         1   929.74  973.74
## - wines                 1   932.02  976.02
## - visits                1   935.75  979.75
## - mar_stat_Widow        1   935.89  979.89
## - cmp3_X1               1   937.63  981.63
## - mar_stat_Divorce      1   939.48  983.48
## - mar_stat_Single       1   945.22  989.22
## - cmp1_X1               1   947.36  991.36
## - store                 1   949.15  993.15
## - teens                 1   949.96  993.96
## - recency               1   988.91 1032.91
##
## Step:  AIC=963.71
## response ~ teens + recency + wines + fruits + meat + fish + gold +
##     deals + web + catalog + store + visits + education_Graduation +
##     education_Master + education_PhD + mar_stat_Divorce + mar_stat_Single +
##     mar_stat_Widow + mar_stat_YOLO + cmp3_X1 + cmp1_X1
##
##                        Df Deviance     AIC
## - catalog               1   920.68  962.68
## - fish                  1   920.91  962.91
## - education_Graduation  1   921.14  963.14
## <none>                      919.71  963.71
## - mar_stat_YOLO         1   922.47  964.47
## - fruits                1   922.61  964.61
## + income                1   919.05  965.05
## + education_Basic       1   919.09  965.09
## + cmplain_X1            1   919.21  965.21
## + mar_stat_Partner      1   919.39  965.39
## + mar_stat_Alone        1   919.42  965.42
## + mar_stat_Married      1   919.46  965.46
## + age                   1   919.52  965.52
## + kids                  1   919.65  965.65
## + sweets                1   919.66  965.66
## - web                   1   924.51  966.51
## - deals                 1   924.76  966.76
## - education_Master      1   924.90  966.90
## - gold                  1   925.89  967.89
## - education_PhD         1   931.18  973.18
## - meat                  1   932.68  974.68
## - wines                 1   935.10  977.10
## - mar_stat_Widow        1   936.29  978.29
## - visits                1   936.99  978.99
```

```
## - cmp3_X1               1   938.87  980.87
## - mar_stat_Divorce      1   940.04  982.04
## - mar_stat_Single       1   945.66  987.66
## - store                 1   949.17  991.17
## - cmp1_X1               1   949.76  991.76
## - teens                 1   950.29  992.29
## - recency               1   990.69 1032.69
##
## Step:  AIC=962.68
## response ~ teens + recency + wines + fruits + meat + fish + gold +
##     deals + web + store + visits + education_Graduation + education_Master +
##     education_PhD + mar_stat_Divorce + mar_stat_Single + mar_stat_Widow +
##     mar_stat_YOLO + cmp3_X1 + cmp1_X1
##
##                         Df Deviance     AIC
## - fish                   1   921.69  961.69
## - education_Graduation   1   922.07  962.07
## <none>                       920.68  962.68
## - fruits                 1   923.36  963.36
## - mar_stat_YOLO          1   923.37  963.37
## + catalog                1   919.71  963.71
## + income                 1   919.75  963.75
## + education_Basic        1   920.04  964.04
## + cmplain_X1             1   920.20  964.20
## + mar_stat_Partner       1   920.36  964.36
## + mar_stat_Alone         1   920.37  964.37
## + mar_stat_Married       1   920.43  964.43
## + age                    1   920.53  964.53
## + sweets                 1   920.62  964.62
## + kids                   1   920.68  964.68
## - web                    1   925.65  965.65
## - education_Master       1   925.67  965.67
## - deals                  1   926.77  966.77
## - gold                   1   927.55  967.55
## - education_PhD          1   932.32  972.32
## - visits                 1   937.00  977.00
## - mar_stat_Widow         1   937.32  977.32
## - wines                  1   938.41  978.41
## - cmp3_X1                1   940.32  980.32
## - meat                   1   940.45  980.45
## - mar_stat_Divorce       1   941.37  981.37
## - mar_stat_Single        1   946.53  986.53
## - store                  1   951.04  991.04
## - teens                  1   951.07  991.07
## - cmp1_X1                1   951.35  991.35
## - recency                1   991.79 1031.79
##
```

```
## Step:  AIC=961.69
## response ~ teens + recency + wines + fruits + meat + gold + deals +
##     web + store + visits + education_Graduation + education_Master +
##     education_PhD + mar_stat_Divorce + mar_stat_Single + mar_stat_Widow +
##     mar_stat_YOLO + cmp3_X1 + cmp1_X1
##
##                         Df Deviance     AIC
## - education_Graduation   1   923.18  961.18
## - fruits                 1   923.66  961.66
## <none>                       921.69  961.69
## - mar_stat_YOLO          1   924.38  962.38
## + fish                   1   920.68  962.68
## + income                 1   920.79  962.79
## + catalog                1   920.91  962.91
## + education_Basic        1   921.10  963.10
## + cmplain_X1             1   921.21  963.21
## + mar_stat_Alone         1   921.39  963.39
## + mar_stat_Partner       1   921.42  963.42
## + age                    1   921.47  963.47
## + sweets                 1   921.47  963.47
## + mar_stat_Married       1   921.49  963.49
## + kids                   1   921.69  963.69
## - web                    1   926.56  964.56
## - education_Master       1   927.32  965.32
## - gold                   1   927.85  965.85
## - deals                  1   928.17  966.17
## - education_PhD          1   934.25  972.25
## - mar_stat_Widow         1   938.09  976.09
## - visits                 1   939.29  977.29
## - wines                  1   939.37  977.37
## - meat                   1   940.63  978.63
## - cmp3_X1                1   941.95  979.95
## - mar_stat_Divorce       1   942.21  980.21
## - mar_stat_Single        1   947.61  985.61
## - teens                  1   951.80  989.80
## - cmp1_X1                1   951.87  989.87
## - store                  1   953.55  991.55
## - recency                1   992.82 1030.82
##
## Step:  AIC=961.18
## response ~ teens + recency + wines + fruits + meat + gold + deals +
##     web + store + visits + education_Master + education_PhD +
##     mar_stat_Divorce + mar_stat_Single + mar_stat_Widow + mar_stat_YOLO +
##     cmp3_X1 + cmp1_X1
##
##                         Df Deviance     AIC
## - fruits                 1   925.16  961.16
```

```
## <none>                      923.18  961.18
## + education_Basic       1   921.66  961.66
## + education_Graduation  1   921.69  961.69
## - mar_stat_YOLO         1   925.88  961.88
## + fish                  1   922.07  962.07
## + income                1   922.08  962.08
## + catalog               1   922.44  962.44
## + cmplain_X1            1   922.80  962.80
## + mar_stat_Partner      1   922.87  962.87
## + mar_stat_Alone        1   922.88  962.88
## + sweets                1   922.93  962.93
## + mar_stat_Married      1   922.95  962.95
## + age                   1   923.04  963.04
## + kids                  1   923.17  963.17
## - education_Master      1   927.91  963.91
## - web                   1   928.18  964.18
## - gold                  1   929.37  965.37
## - deals                 1   929.70  965.70
## - education_PhD         1   939.60  975.60
## - mar_stat_Widow        1   939.91  975.91
## - visits                1   940.54  976.54
## - wines                 1   941.55  977.55
## - meat                  1   943.03  979.03
## - cmp3_X1               1   943.28  979.28
## - mar_stat_Divorce      1   943.72  979.72
## - mar_stat_Single       1   949.36  985.36
## - teens                 1   952.43  988.43
## - cmp1_X1               1   953.11  989.11
## - store                 1   955.78  991.78
## - recency               1   994.26 1030.26
##
## Step:  AIC=961.16
## response ~ teens + recency + wines + meat + gold + deals + web +
##     store + visits + education_Master + education_PhD + mar_stat_Divorce +
##     mar_stat_Single + mar_stat_Widow + mar_stat_YOLO + cmp3_X1 +
##     cmp1_X1
##
##                        Df Deviance     AIC
## <none>                      925.16  961.16
## + fruits                1   923.18  961.18
## + education_Basic       1   923.61  961.61
## + education_Graduation  1   923.66  961.66
## - mar_stat_YOLO         1   927.87  961.87
## + income                1   923.87  961.87
## + catalog               1   924.53  962.53
## + cmplain_X1            1   924.78  962.78
## + fish                  1   924.80  962.80
```

```
## + mar_stat_Partner      1   924.84  962.84
## + mar_stat_Alone        1   924.85  962.85
## + mar_stat_Married      1   924.92  962.92
## + age                   1   925.04  963.04
## + sweets                1   925.14  963.14
## + kids                  1   925.16  963.16
## - education_Master      1   929.47  963.47
## - web                   1   930.22  964.22
## - deals                 1   931.08  965.08
## - gold                  1   933.23  967.23
## - education_PhD         1   940.54  974.54
## - visits                1   941.39  975.39
## - mar_stat_Widow        1   941.80  975.80
## - wines                 1   944.01  978.01
## - cmp3_X1               1   944.22  978.22
## - mar_stat_Divorce      1   945.94  979.94
## - meat                  1   949.91  983.91
## - mar_stat_Single       1   951.16  985.16
## - cmp1_X1               1   955.32  989.32
## - teens                 1   955.86  989.86
## - store                 1   956.36  990.36
## - recency               1   995.49 1029.49
```

```
summary(step)
```

```
##
## Call:
## glm(formula = response ~ teens + recency + wines + meat + gold +
##     deals + web + store + visits + education_Master + education_PhD +
##     mar_stat_Divorce + mar_stat_Single + mar_stat_Widow + mar_stat_YOLO +
##     cmp3_X1 + cmp1_X1, family = binomial(link = "logit"), data = bake_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4838  -0.4773  -0.2916  -0.1484   3.1988
##
## Coefficients:
##                     Estimate  Std. Error z value            Pr(>|z|)
## (Intercept)       -2.9481838   0.4093830  -7.202 0.00000000000059541 ***
## teens             -1.1138877   0.2111962  -5.274 0.00000013334711137 ***
## recency           -0.0255378   0.0032317  -7.902 0.00000000000000274 ***
## wines              0.0015298   0.0003534   4.328 0.00001501684481894 ***
## meat               0.0022493   0.0004565   4.927 0.00000083340103793 ***
## gold               0.0048602   0.0016927   2.871              0.00409 **
## deals              0.1191855   0.0491762   2.424              0.01537 *
## web                0.0761367   0.0346245   2.199              0.02788 *
## store             -0.2036471   0.0384618  -5.295 0.00000011915791150 ***
## visits             0.1873605   0.0467095   4.011 0.00006041463788527 ***
## education_Master   0.5002172   0.2369329   2.111              0.03475 *
## education_PhD      0.8437063   0.2130747   3.960 0.00007505234068219 ***
## mar_stat_Divorce   1.2467961   0.2627020   4.746 0.00000207430424697 ***
## mar_stat_Single    1.0448055   0.2023676   5.163 0.00000024314291594 ***
## mar_stat_Widow     1.6615621   0.3791259   4.383 0.00001172643099335 ***
## mar_stat_YOLO     14.6459736 535.4112254   0.027              0.97818
## cmp3_X1            2.4362115   0.5887482   4.138 0.00003504201041675 ***
## cmp1_X1            1.5396812   0.2798718   5.501 0.00000003768276905 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1317.96  on 1567  degrees of freedom
## Residual deviance:  925.16  on 1550  degrees of freedom
## AIC: 961.16
##
## Number of Fisher Scoring iterations: 12
```

Use tidymodel framework to fit

```
uni_steprecipe <- recipe(response ~ teens + recency + wines + meat + gold + deals + web + store + visits + e
          ducation + education + mar_stat + cmp3 + cmp1, data = train) %>%
    step_impute_median(all_numeric()) %>%
    prep()

uni_steprecipe
```

```
## Recipe
##
## Inputs:
##
##       role #variables
##    outcome          1
##  predictor         13
##
## Training data contained 1568 data points and no missing data.
##
## Operations:
##
## Median imputation for teens, recency, wines, meat, gold, deals, web, ... [trained]
```

```
# apply new recipe
bake_steptrain <- bake(uni_steprecipe, new_data = train)
bake_steptest  <- bake(uni_steprecipe, new_data = test)

logistic_step <- logistic_reg(mode = "classification") %>%
                 set_engine("glm") %>%
                 fit(response ~ ., data = bake_steptrain)



## check out your parameter estimates ...
tidy(logistic_step) %>%
  mutate_at(c("estimate", "std.error", "statistic", "p.value"), round, 4)
```

term	estimate	std.error	statistic	p.value
<chr>	<dbl>	<dbl>	<dbl>	<dbl>
(Intercept)	-16.0534	882.7436	-0.0182	0.9855
teens	-1.1444	0.2118	-5.4042	0.0000
recency	-0.0256	0.0032	-7.9055	0.0000
wines	0.0015	0.0004	4.2733	0.0000

term	estimate	std.error	statistic	p.value
<chr>	<dbl>	<dbl>	<dbl>	<dbl>
meat	0.0022	0.0005	4.7964	0.0000
gold	0.0049	0.0017	2.8640	0.0042
deals	0.1208	0.0491	2.4575	0.0140
web	0.0735	0.0348	2.1109	0.0348
store	-0.2036	0.0385	-5.2830	0.0000
visits	0.1891	0.0466	4.0557	0.0000

1-10 of 23 rows Previous **1** 2 3 Next

```
# training predictions from stepwise model
predict(logistic_step, bake_steptrain, type = "prob") %>%
  bind_cols(.,predict(logistic_step, bake_steptrain)) %>%
  bind_cols(.,bake_steptrain) -> scored_train_step

head(scored_train_step)
```

.pred_0	.pred_1	.pred_class	teens	recency	wines	meat	gold	deals	w...
<dbl>	<dbl>	<fct>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
0.8841453	0.115854713	0	0	25	4	11	15	2	2
0.8419604	0.158039586	0	0	52	625	169	35	1	4
0.9913387	0.008661273	0	0	82	1	3	7	1	1
0.9654285	0.034571533	0	0	71	6	9	35	4	2
0.2731816	0.726818439	1	0	12	912	170	97	1	11
0.8766419	0.123358083	0	1	8	216	6	9	2	5

6 rows | 1-10 of 17 columns

```
# testing predictions from stepwise model
predict(logistic_step, bake_steptest, type = "prob") %>%
  bind_cols(.,predict(logistic_step, bake_steptest)) %>%
  bind_cols(.,bake_steptest) -> scored_test_step

head(scored_test_step)
```

.pred_0 <dbl>	.pred_1 <dbl>	.pred_class <fct>	teens <dbl>	recency <dbl>	wines <dbl>	meat <dbl>	gold <dbl>	deals <dbl>	w... <dbl>
0.3206521	0.67934792	1	0	58	635	546	88	3	8
0.9475045	0.05249551	0	0	26	426	127	42	1	8
0.9344891	0.06551095	0	0	26	11	20	5	2	2
0.6813872	0.31861279	0	0	19	14	24	2	1	3
0.6864405	0.31355947	0	1	68	28	6	13	1	1
0.9821763	0.01782366	0	0	59	6	11	16	1	2

6 rows | 1-10 of 17 columns

Evaulate Reduced Regression Model

```
# Evaluate Stepwise Model
# AUC: Train and Test
options(yardstick.event_first = FALSE)

scored_train_step %>%
  metrics(response, .pred_1, estimate = .pred_class) %>%
  mutate(part="training") %>%
  bind_rows(scored_test_step %>%
              metrics(response, .pred_1, estimate = .pred_class) %>%
              mutate(part="testing")
  ) %>%
  filter(.metric %in% c("accuracy", "roc_auc"))
```

.metric <chr>	.estimator <chr>	.estimate <dbl>	part <chr>
accuracy	binary	0.8864796	training
roc_auc	binary	0.8667486	training
accuracy	binary	0.8750000	testing
roc_auc	binary	0.8158155	testing

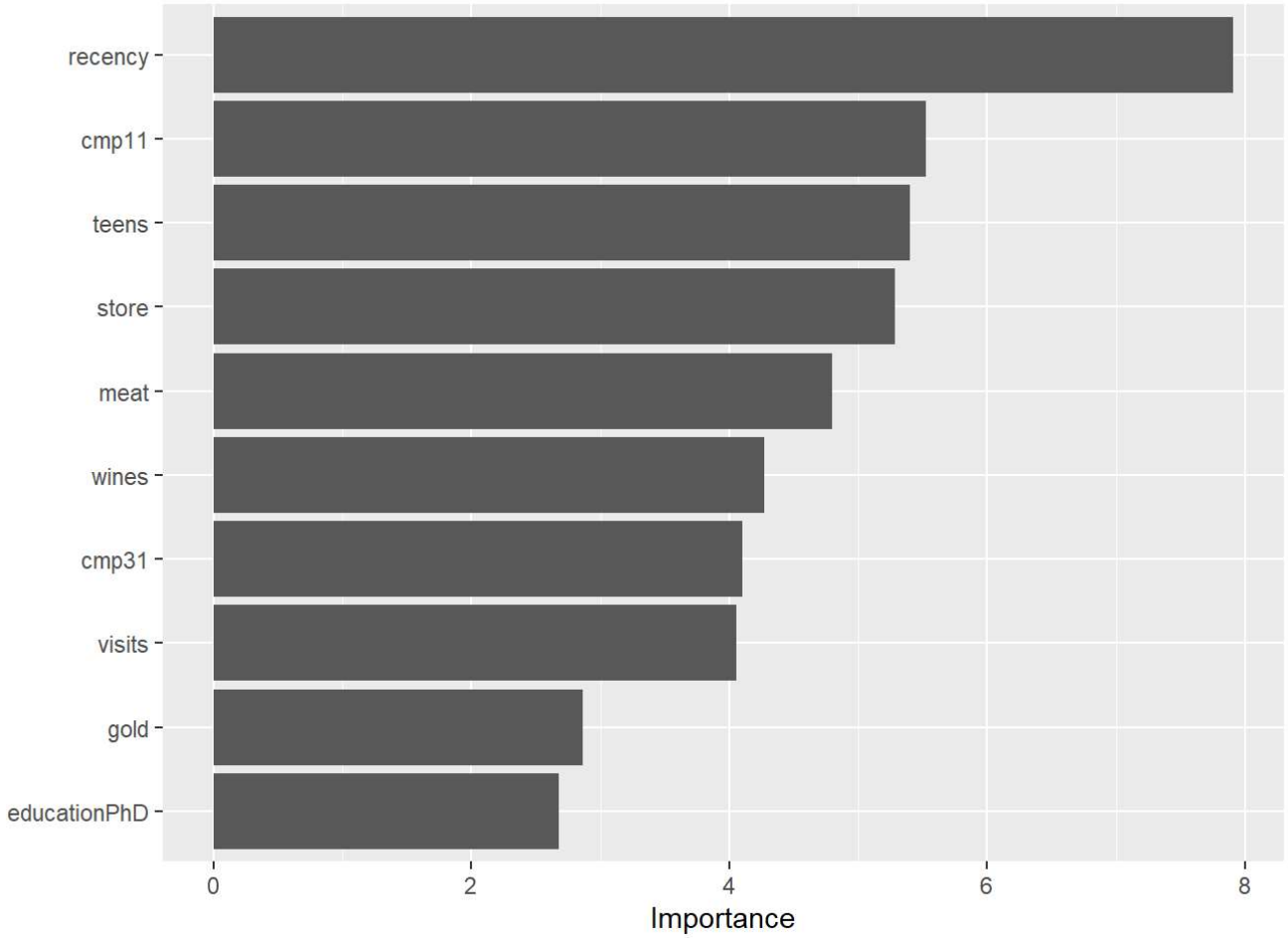
4 rows

```
model_glm <- glm(response ~ teens + recency + wines + meat + gold + deals + web + store + visits + education
          + education + mar_stat + cmp3 + cmp1, data = train, family=binomial(link="logit"))


# Variable Importance top 10 features
model_glm %>%
  vi()
```

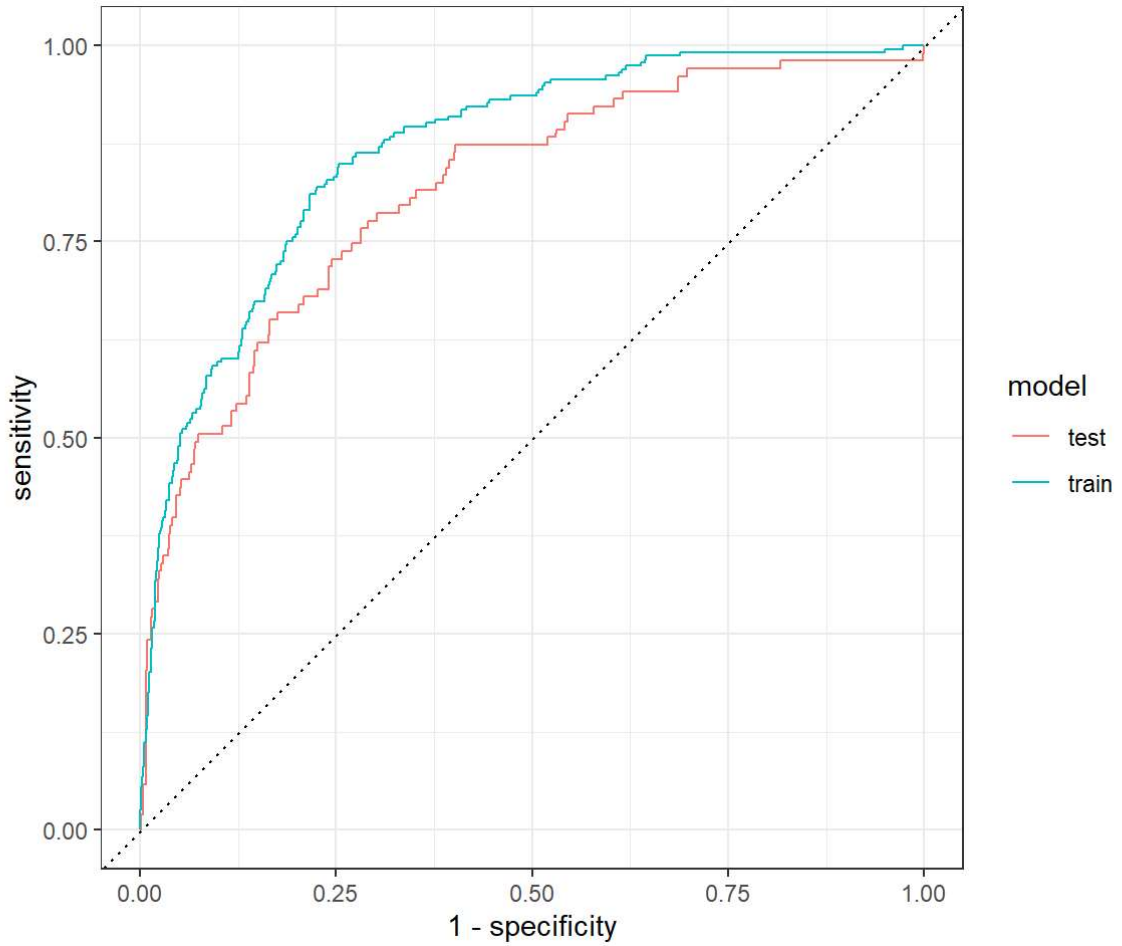
Variable	Importance	Sign
<chr>	<dbl>	<chr>
recency	7.905489138	NEG
cmp11	5.524362758	POS
teens	5.404150680	NEG
store	5.283031442	NEG
meat	4.796351693	POS
wines	4.273318558	POS
cmp31	4.104440089	POS
visits	4.055700617	POS
gold	2.863989725	POS
educationPhD	2.675882288	POS

1-10 of 22 rows Previous **1** 2 3 Next

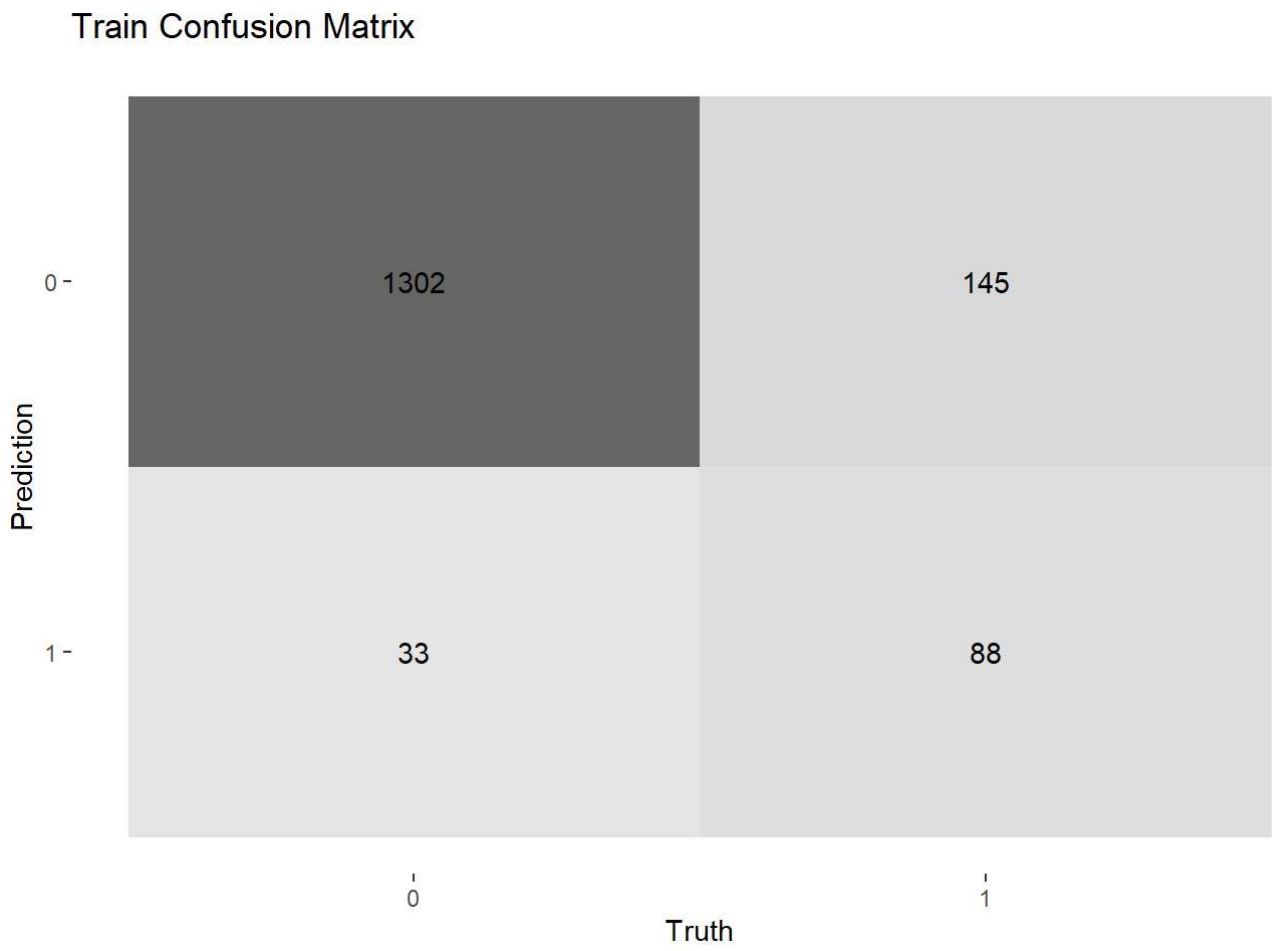
```
model_glm %>%
  vip(num_features = 10)
```



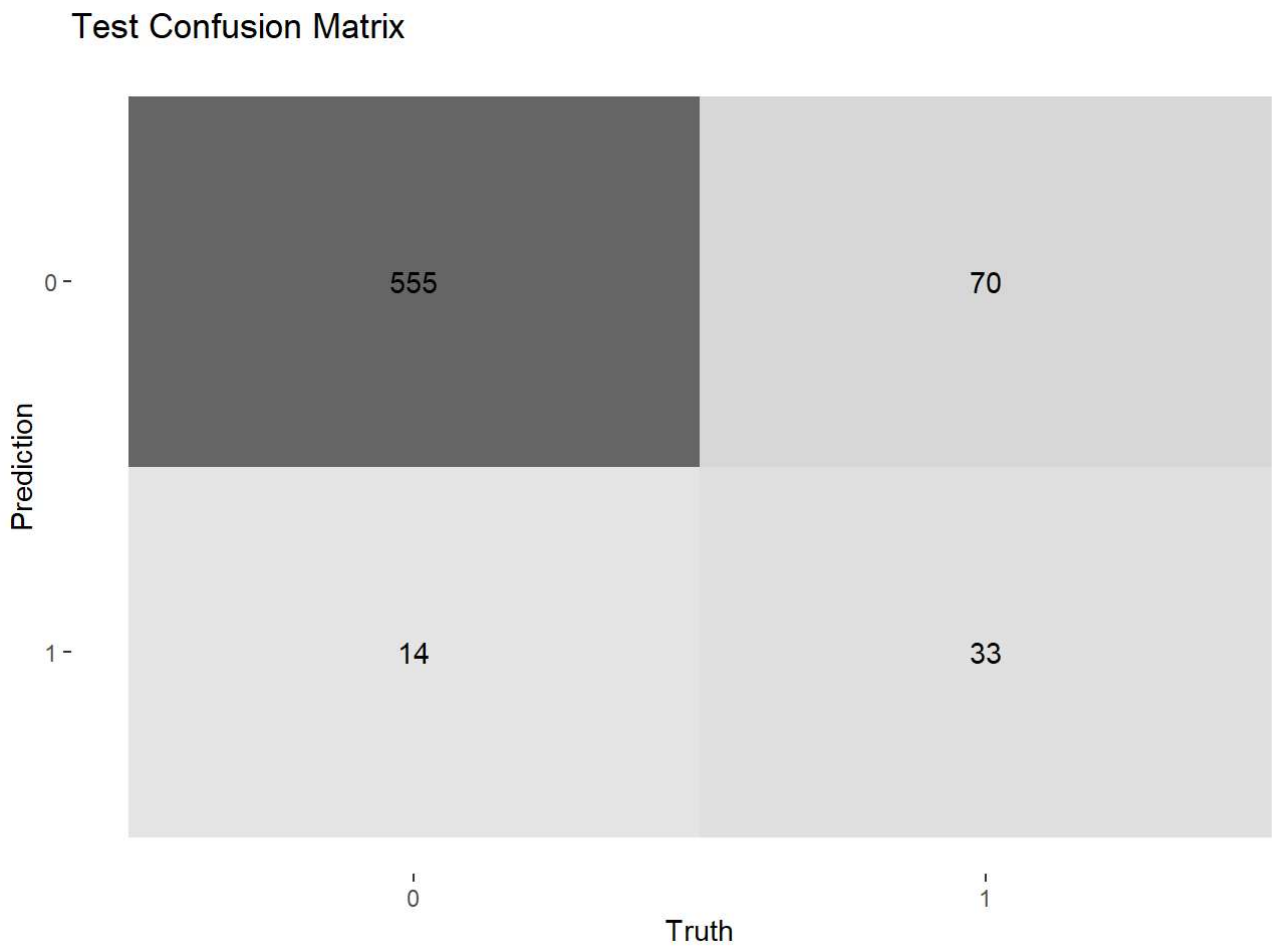

```
# ROC Charts
scored_train_step %>%
  mutate(model = "train") %>%
  bind_rows(scored_test_step %>%
              mutate(model="test")) %>%
  group_by(model) %>%
  roc_curve(response, .pred_1) %>%
  autoplot()
```




```
# Confusion Matrices
scored_train_step %>%
  conf_mat(response, .pred_class) %>%
  autoplot(type = "heatmap") +
  labs(title="Train Confusion Matrix")
```




```
scored_test_step %>%
  conf_mat(response, .pred_class) %>%
  autoplot(type = "heatmap") +
  labs(title="Test Confusion Matrix")
```



Define Recipe & Bake

```
recipe <- recipe(response ~ teens + recency + wines + meat + gold + deals + web + store + visits + education
          + education + mar_stat + cmp3 + cmp1, data=train) %>%
    step_impute_median(all_numeric_predictors()) %>%
    step_unknown(all_nominal_predictors()) %>%
    step_scale(all_numeric_predictors()) %>%
    step_novel(all_nominal_predictors()) %>% # new factor levels
    step_dummy(all_nominal_predictors(), one_hot = TRUE) %>%
    step_nzv(all_predictors()) %>%
    prep()


recipe
```

```
## Recipe
##
## Inputs:
##
##       role #variables
##    outcome          1
##  predictor         13
##
## Training data contained 1568 data points and no missing data.
##
## Operations:
##
## Median imputation for teens, recency, wines, meat, gold, deals, web, ... [trained]
## Unknown factor level assignment for education, mar_stat, cmp3, cmp1 [trained]
## Scaling for teens, recency, wines, meat, gold, deals, web, ... [trained]
## Novel factor level assignment for education, mar_stat, cmp3, cmp1 [trained]
## Dummy variables from education, mar_stat, cmp3, cmp1 [trained]
## Sparse, unbalanced variable filter removed education_Basic, education_unknown, edu... [trained]
```

```
bake(recipe %>% prep(), train, composition = "tibble") %>% head()
```

teens	recency	wines	meat	gold	deals	web	store	visits	re
<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<1
0.000000	0.8641098	0.011942611	0.04827289	0.2878325	1.0513497	0.7107768	0.9355548	3.3274265	0
0.000000	1.7973484	1.866033023	0.74164706	0.6716093	0.5256749	1.4215536	1.8711096	0.4159283	0
0.000000	2.8342802	0.002985653	0.01316533	0.1343219	0.5256749	0.3553884	0.9355548	3.3274265	0
0.000000	2.4540719	0.017913917	0.03949600	0.6716093	2.1026994	0.7107768	0.9355548	2.9114981	1
0.000000	0.4147727	2.722915388	0.74603550	1.8613172	0.5256749	3.9092725	1.2474064	1.6637132	1
1.837749	0.2765151	0.644901013	0.02633066	0.1726995	1.0513497	1.7769421	1.2474064	2.9114981	0

6 rows | 1-10 of 20 columns

```
bake_train <- bake(recipe, new_data = train)
bake_test  <- bake(recipe, new_data = test)
```

Define KNN Model

```
knn_model <- nearest_neighbor(neighbors = 7) %>%
    set_mode("classification") %>%
    set_engine("kknn")
```

KNN Workflow

```
knn_workflow <- workflow() %>%
    add_recipe(recipe) %>%
    add_model(knn_model) %>%
    fit(train)
```

Score KNN model

```
# score training
scored_train_knn <- predict(knn_workflow, train, type="prob") %>%
    bind_cols(predict(knn_workflow, train, type="class")) %>%
    bind_cols(.,train)
# score testing
scored_test_knn <- predict(knn_workflow, test, type="prob") %>%
    bind_cols(predict(knn_workflow, test, type="class")) %>%
    bind_cols(.,test)
```

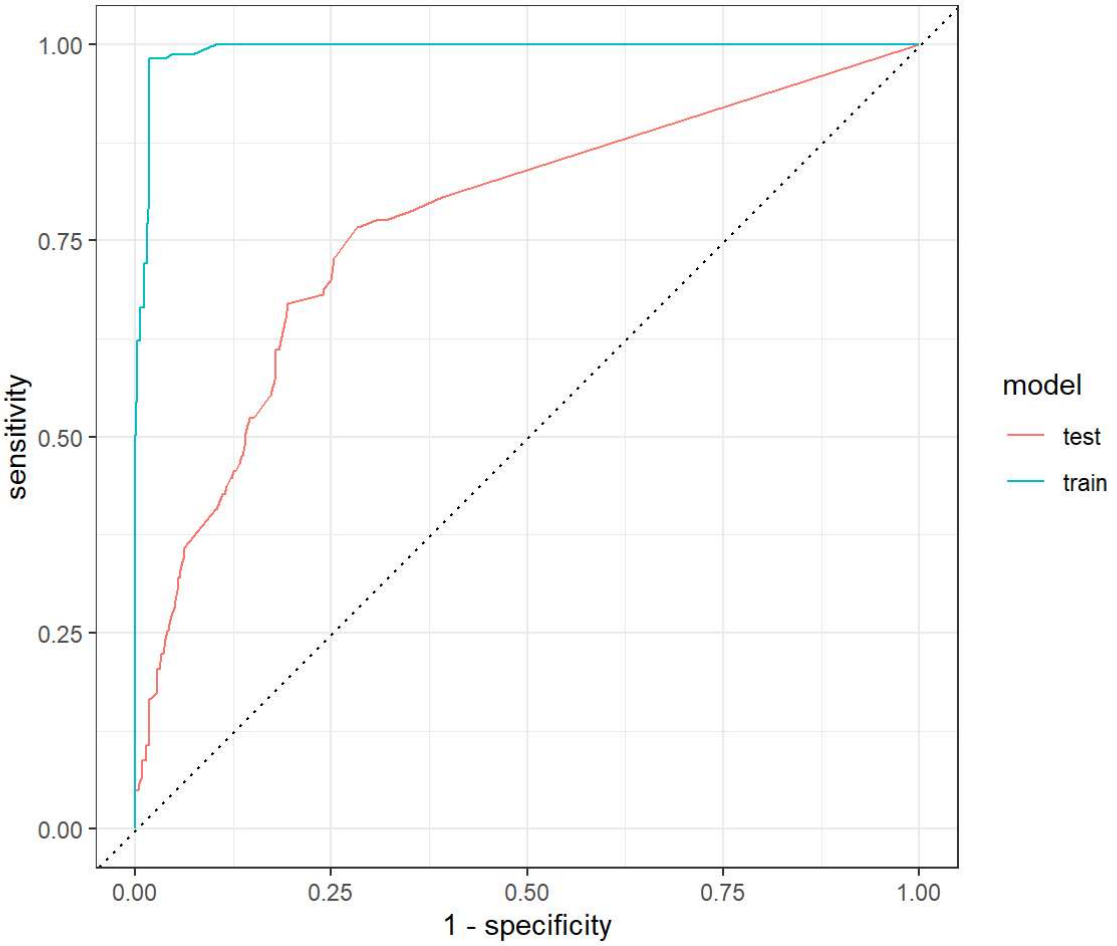
Evaluate (KNN = 7)

```
options(yardstick.event_first = FALSE)
# Metrics: Train and Test
scored_train_knn %>%
    metrics(response, .pred_1, estimate = .pred_class) %>%
    mutate(part="training") %>%
    bind_rows(scored_test_knn %>%
                metrics(response, .pred_1, estimate = .pred_class) %>%
                mutate(part="testing") ) %>%
    filter(.metric %in% c('accuracy','roc_auc')) %>%
    pivot_wider(names_from = .metric, values_from=.estimate)
```

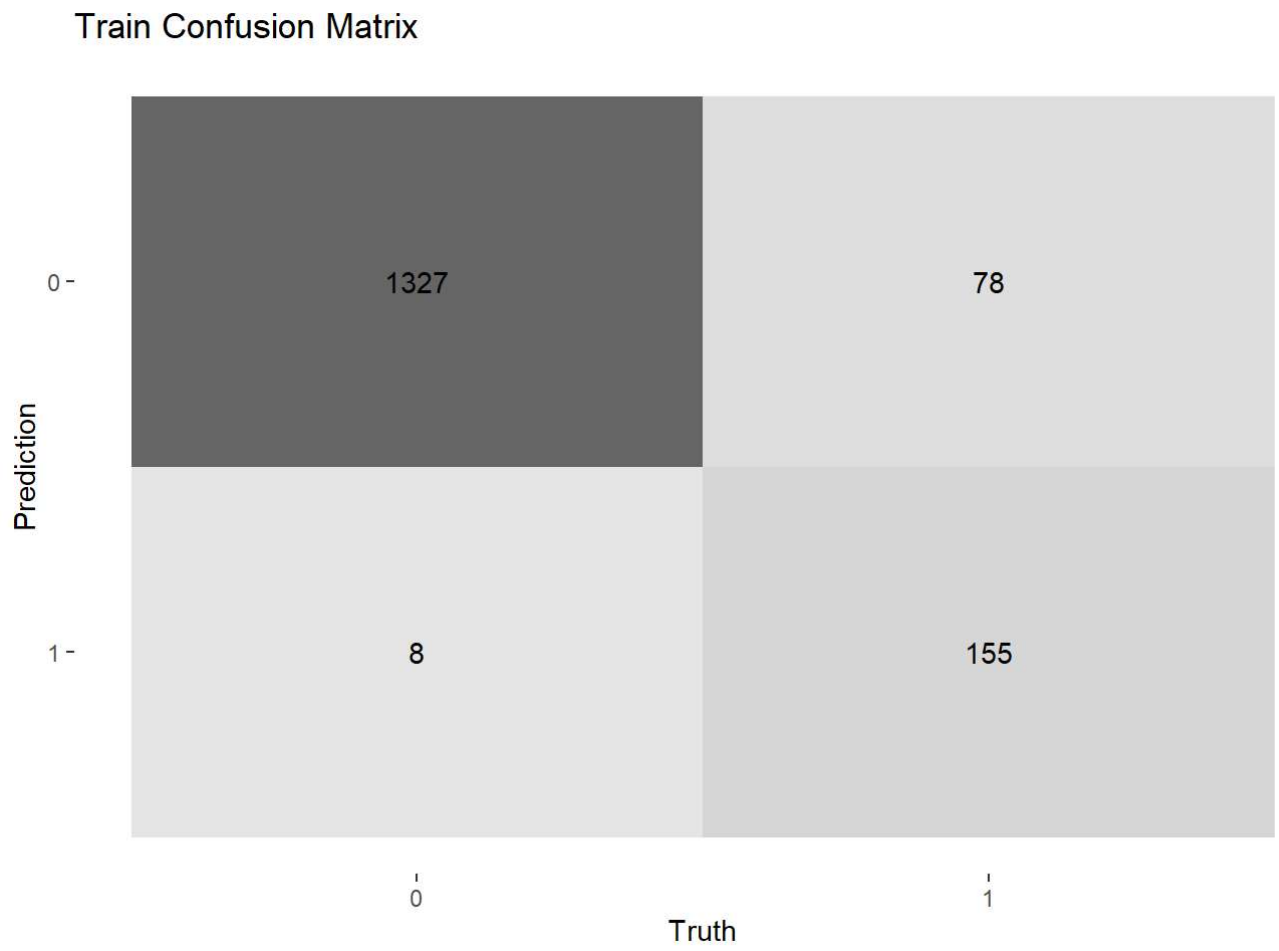
.estimator	part	accuracy	roc_auc
<chr>	<chr>	<dbl>	<dbl>
binary	training	0.9451531	0.9931588
binary	testing	0.8482143	0.7719300

2 rows

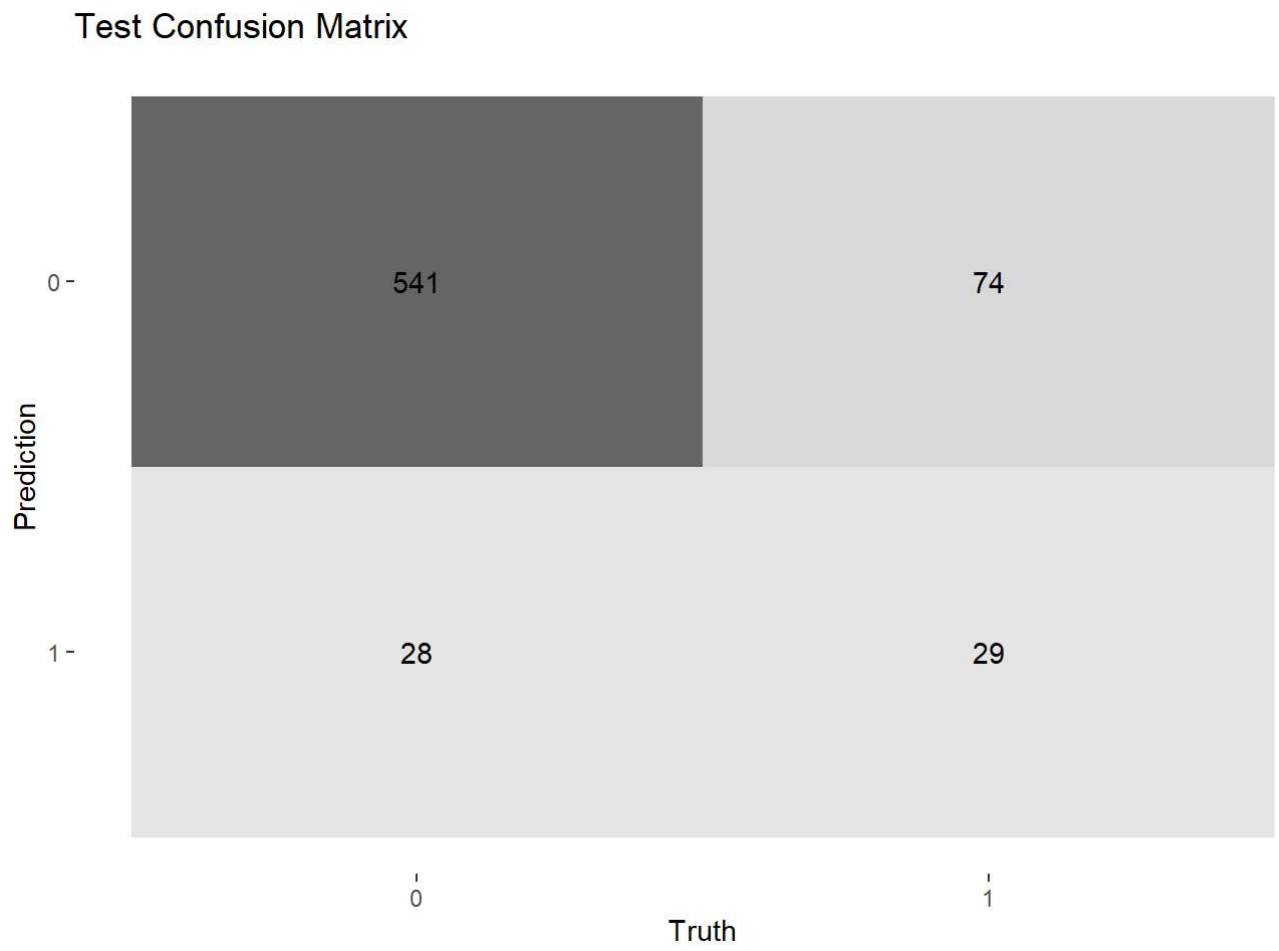
```
# ROC Charts
scored_train_knn %>%
  mutate(model = "train") %>%
  bind_rows(scored_test_knn %>%
              mutate(model="test")) %>%
  group_by(model) %>%
  roc_curve(response, .pred_1) %>%
  autoplot()
```




```
scored_train_knn %>%
  conf_mat(response, .pred_class) %>%
  autoplot(type = "heatmap") +
  labs(title="Train Confusion Matrix")
```




```
scored_test_knn %>%
  conf_mat(response, .pred_class) %>%
  autoplot(type = "heatmap") +
  labs(title="Test Confusion Matrix")
```

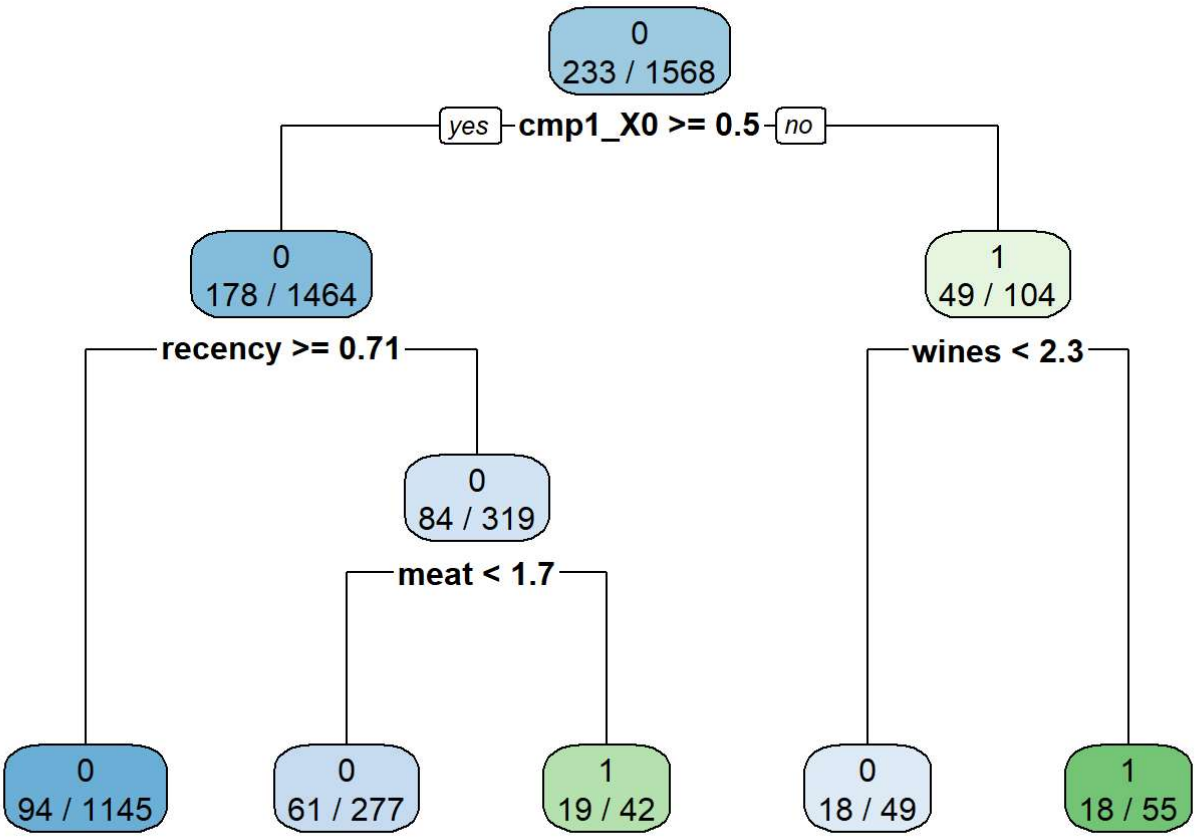


Define Decision Tree Model

```
retail_tree <- decision_tree(mode="classification",
                             cost_complexity = 0.001,
                             tree_depth = 5,
                             min_n = 100) %>%
                set_engine("rpart") %>%
                fit(response ~ ., data=bake_train)

retail_tree$fit
```

```
## n= 1568
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 1568 233 0 (0.85140306 0.14859694)
##    2) cmp1_X0>=0.5 1464 178 0 (0.87841530 0.12158470)
##      4) recency>=0.7085701 1145  94 0 (0.91790393 0.08209607) *
##      5) recency< 0.7085701 319  84 0 (0.73667712 0.26332288)
##       10) meat< 1.742212 277  61 0 (0.77978339 0.22021661) *
##       11) meat>=1.742212 42  19 1 (0.45238095 0.54761905) *
##    3) cmp1_X0< 0.5 104  49 1 (0.47115385 0.52884615)
##      6) wines< 2.273575 49  18 0 (0.63265306 0.36734694) *
##      7) wines>=2.273575 55  18 1 (0.32727273 0.67272727) *
```

```
options(scipen = 0)


rpart.plot(retail_tree$fit, roundint=FALSE, extra=3)
```



Decision Tree Evaluation

```
# training
predict(retail_tree, bake_train, type = "prob") %>%
  bind_cols(.,predict(retail_tree, bake_train)) %>%
  bind_cols(.,bake_train) -> scored_train_tree

head(scored_train_tree)
```

.pred_0 <dbl>	.pred_1 <dbl>	.pred_class <fct>	teens <dbl>	recency <dbl>	wines <dbl>	meat <dbl>	gold <dbl>	deal <dbl
0.9179039	0.08209607	0	0.000000	0.8641098	0.011942611	0.04827289	0.2878325	1.051349
0.9179039	0.08209607	0	0.000000	1.7973484	1.866033023	0.74164706	0.6716093	0.525674
0.9179039	0.08209607	0	0.000000	2.8342802	0.002985653	0.01316533	0.1343219	0.525674
0.9179039	0.08209607	0	0.000000	2.4540719	0.017913917	0.03949600	0.6716093	2.102699
0.7797834	0.22021661	0	0.000000	0.4147727	2.722915388	0.74603550	1.8613172	0.525674
0.7797834	0.22021661	0	1.837749	0.2765151	0.644901013	0.02633066	0.1726995	1.051349

6 rows | 1-10 of 23 columns

```
# testing
predict(retail_tree, bake_test, type = "prob") %>%
  bind_cols(.,predict(retail_tree, bake_test)) %>%
  bind_cols(.,bake_test) -> scored_test_tree

head(scored_test_tree)
```

.pred_0 <dbl>	.pred_1 <dbl>	.pred_class <fct>	teens <dbl>	recency <dbl>	wines <dbl>	meat <dbl>	gold <dbl>	deal <dbl
0.9179039	0.08209607	0	0.000000	2.0047348	1.89588955	2.39609049	1.68861762	1.577024
0.9179039	0.08209607	0	0.000000	0.8986742	1.27188811	0.55733240	0.80593114	0.525674
0.9179039	0.08209607	0	0.000000	0.8986742	0.03284218	0.08776888	0.09594418	1.051349
0.7797834	0.22021661	0	0.000000	0.6567235	0.04179914	0.10532266	0.03837767	0.525674
0.9179039	0.08209607	0	1.837749	2.3503787	0.08359828	0.02633066	0.24945488	0.525674
0.9179039	0.08209607	0	0.000000	2.0392992	0.01791392	0.04827289	0.30702139	0.525674

6 rows | 1-10 of 23 columns

Decision Tree Evaluate

```
# AUC: Train and Test
scored_train_tree %>%
  metrics(response, .pred_1, estimate = .pred_class) %>%
  mutate(part="training") %>%
  bind_rows( scored_test_tree %>%
               metrics(response, .pred_1, estimate = .pred_class) %>%
               mutate(part="testing")
  )
```

.metric	.estimator	.estimate	part
<chr>	<chr>	<dbl>	<chr>
accuracy	binary	0.8660714	training
kap	binary	0.3027245	training
mn_log_loss	binary	0.3615823	training
roc_auc	binary	0.7147144	training
accuracy	binary	0.8511905	testing
kap	binary	0.2142740	testing
mn_log_loss	binary	0.3879631	testing
roc_auc	binary	0.6695531	testing

8 rows

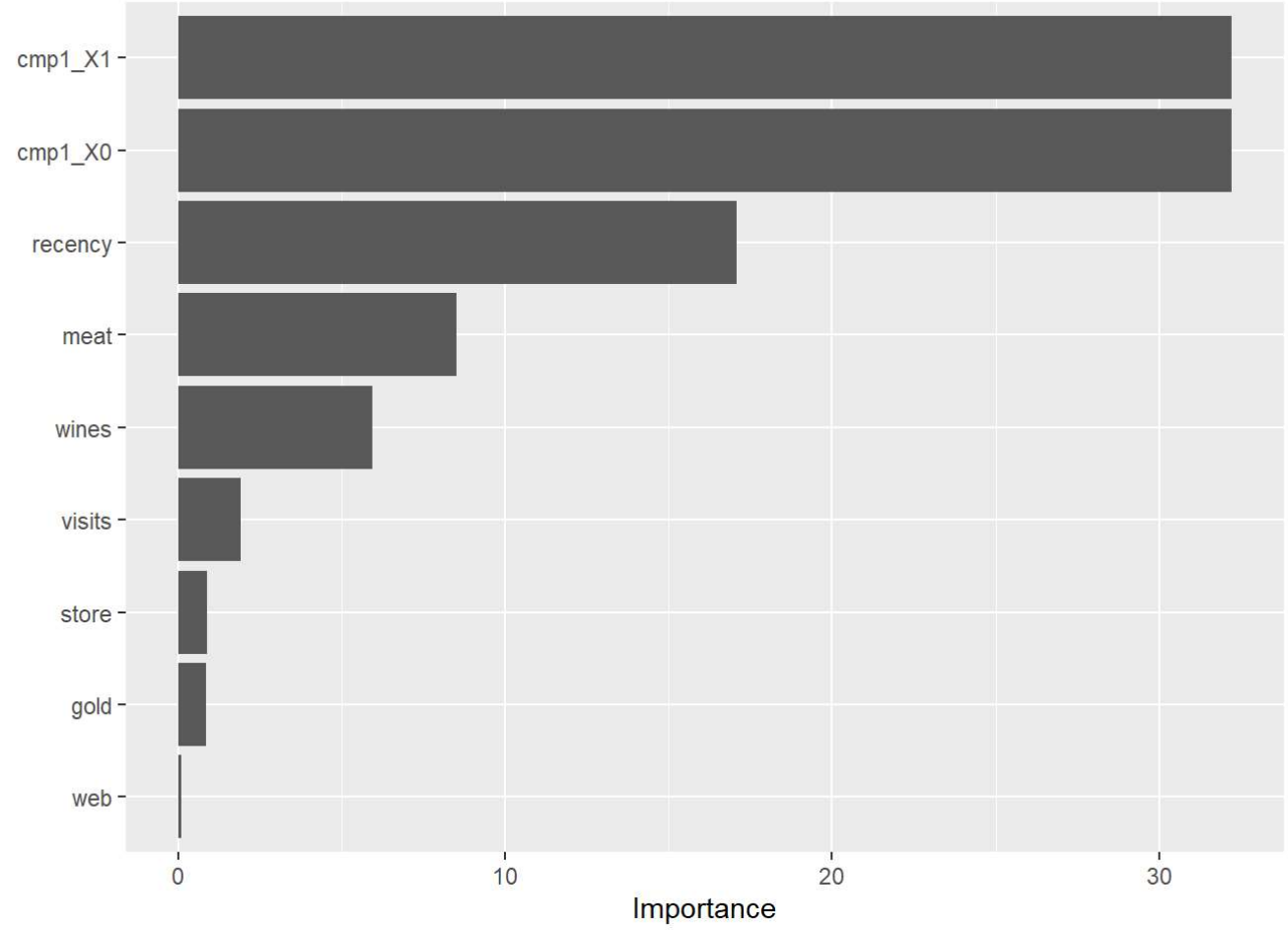
```
# Variable Importance top 10 features
retail_tree %>%
  vi()
```

Variable	Importance
<chr>	<dbl>
cmp1_X0	32.2110565
cmp1_X1	32.2110565
recency	17.0786142
meat	8.5091192

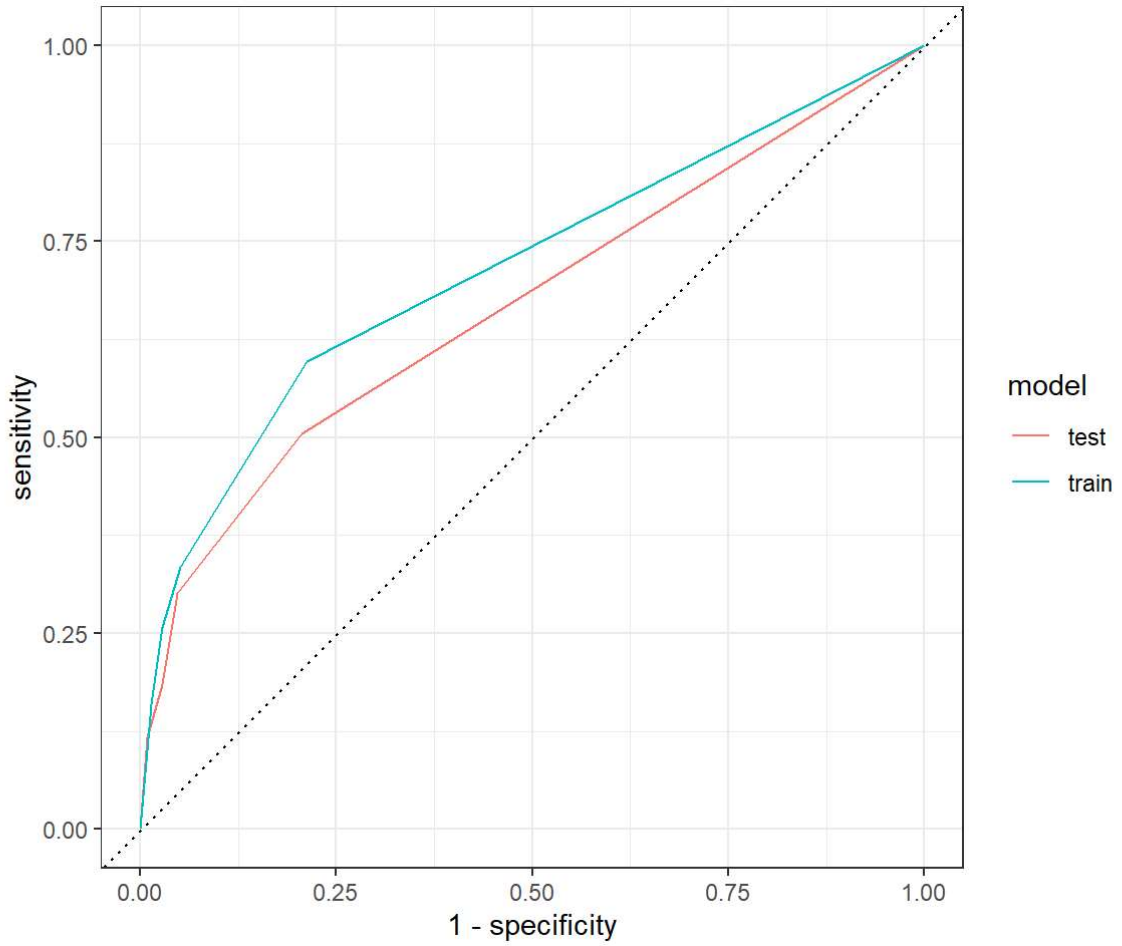
Variable	Importance
<chr>	<dbl>
wines	5.9277385
visits	1.9060500
store	0.8877363
gold	0.8404725
web	0.1027470

9 rows

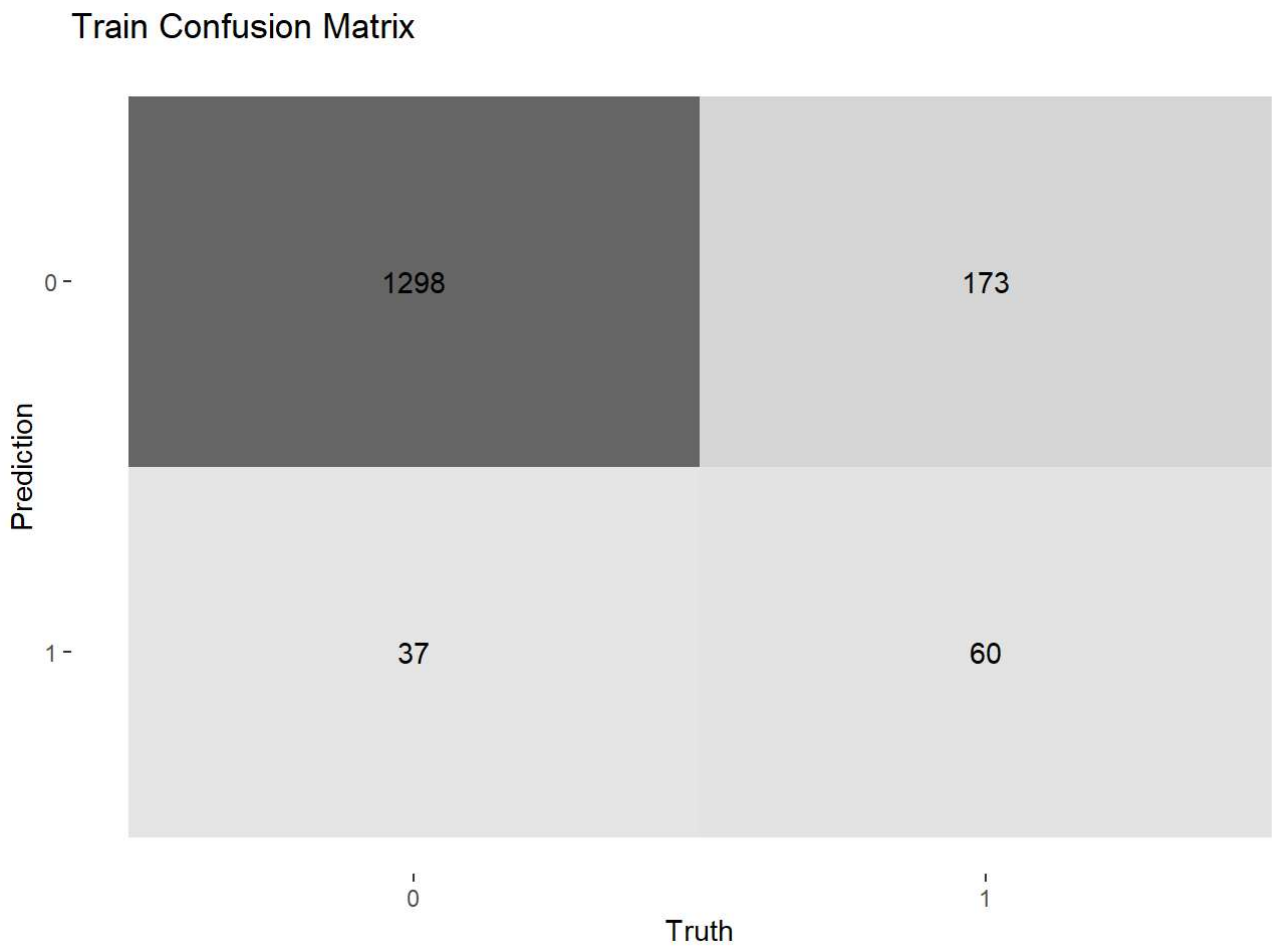
```
retail_tree %>%
  vip(num_features = 10)
```



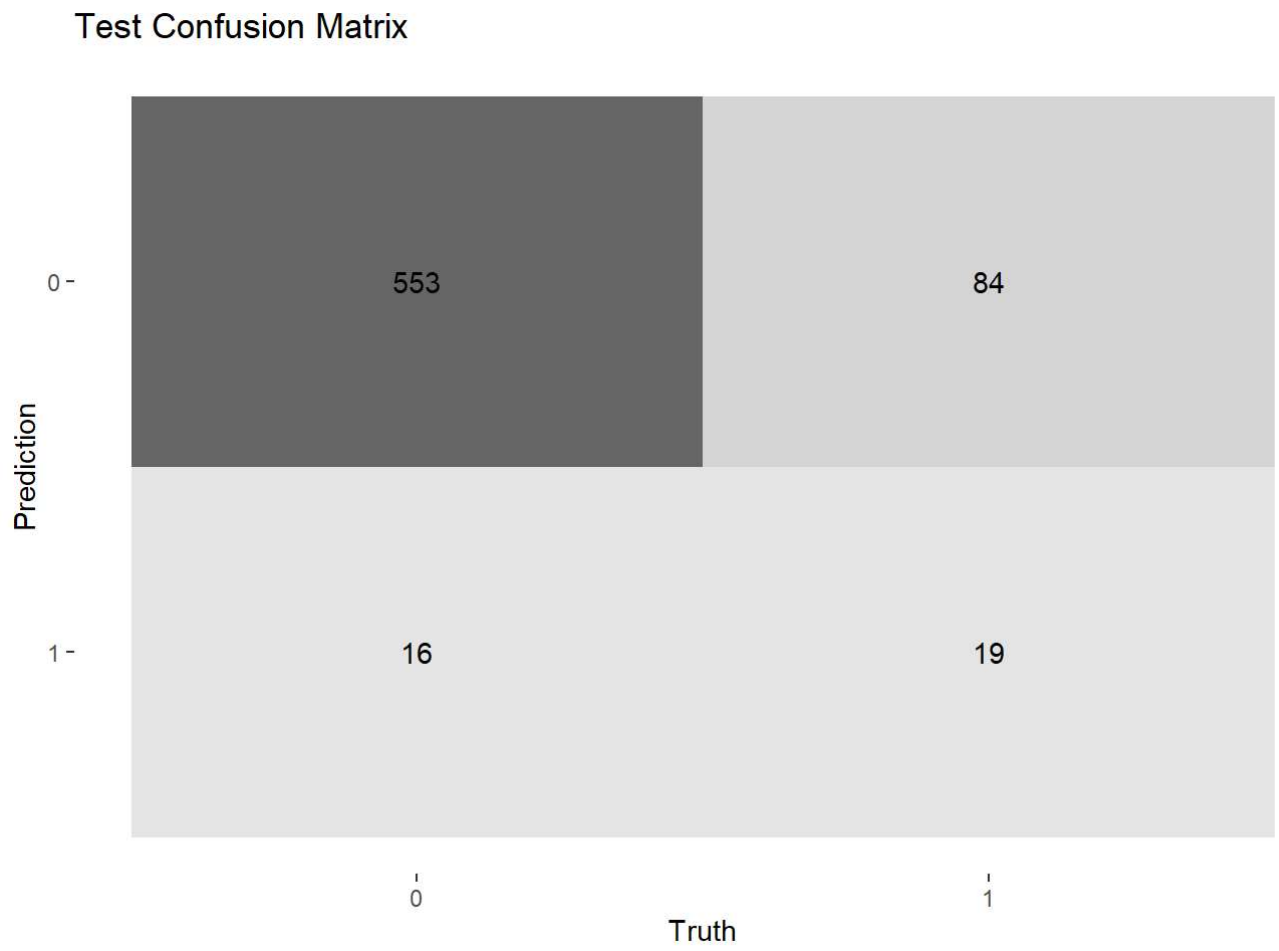
```
# ROC Charts
scored_train_tree %>%
  mutate(model = "train") %>%
  bind_rows(scored_test_tree %>%
              mutate(model="test")) %>%
  group_by(model) %>%
  roc_curve(response, .pred_1) %>%
  autoplot()
```




```
# Confusion Matrices
scored_train_tree %>%
  conf_mat(response, .pred_class) %>%
  autoplot(type = "heatmap") +
  labs(title="Train Confusion Matrix")
```




```
scored_test_tree %>%
  conf_mat(response, .pred_class) %>%
  autoplot( type = "heatmap") +
  labs(title="Test Confusion Matrix")
```



Define Random Forest Model

```
rf_model <- rand_forest(trees=100, min_n = 10) %>%
  set_mode("classification") %>%
  set_engine("ranger", importance="impurity")
```

Random Forest Workflow

```
rf_workflow <- workflow() %>%
  add_recipe(recipe) %>%
  add_model(rf_model) %>%
  fit(train)
```

Score Random Forest Model

```
# score training
predict(rf_workflow, train, type="prob") %>%
  bind_cols(predict(rf_workflow, train, type="class")) %>%
  bind_cols(., train) -> scored_train_rf

# score testing
predict(rf_workflow, test, type="prob") %>%
    bind_cols(predict(rf_workflow, test, type="class")) %>%
    bind_cols(., test) -> scored_test_rf
```

Evaluation (rf_model)

```
options(yardstick.event_first = FALSE)
# Metrics: Train and Test
scored_train_rf %>%
    metrics(response, .pred_1, estimate = .pred_class) %>%
    mutate(part="training") %>%
    bind_rows(scored_test_knn %>%
                metrics(response, .pred_1, estimate = .pred_class) %>%
                mutate(part="testing") ) %>%
    filter(.metric %in% c('accuracy','roc_auc')) %>%
    pivot_wider(names_from = .metric, values_from=.estimate)
```

.estimator	part	accuracy	roc_auc
<chr>	<chr>	<dbl>	<dbl>
binary	training	0.9661990	0.9981659
binary	testing	0.8482143	0.7719300

2 rows

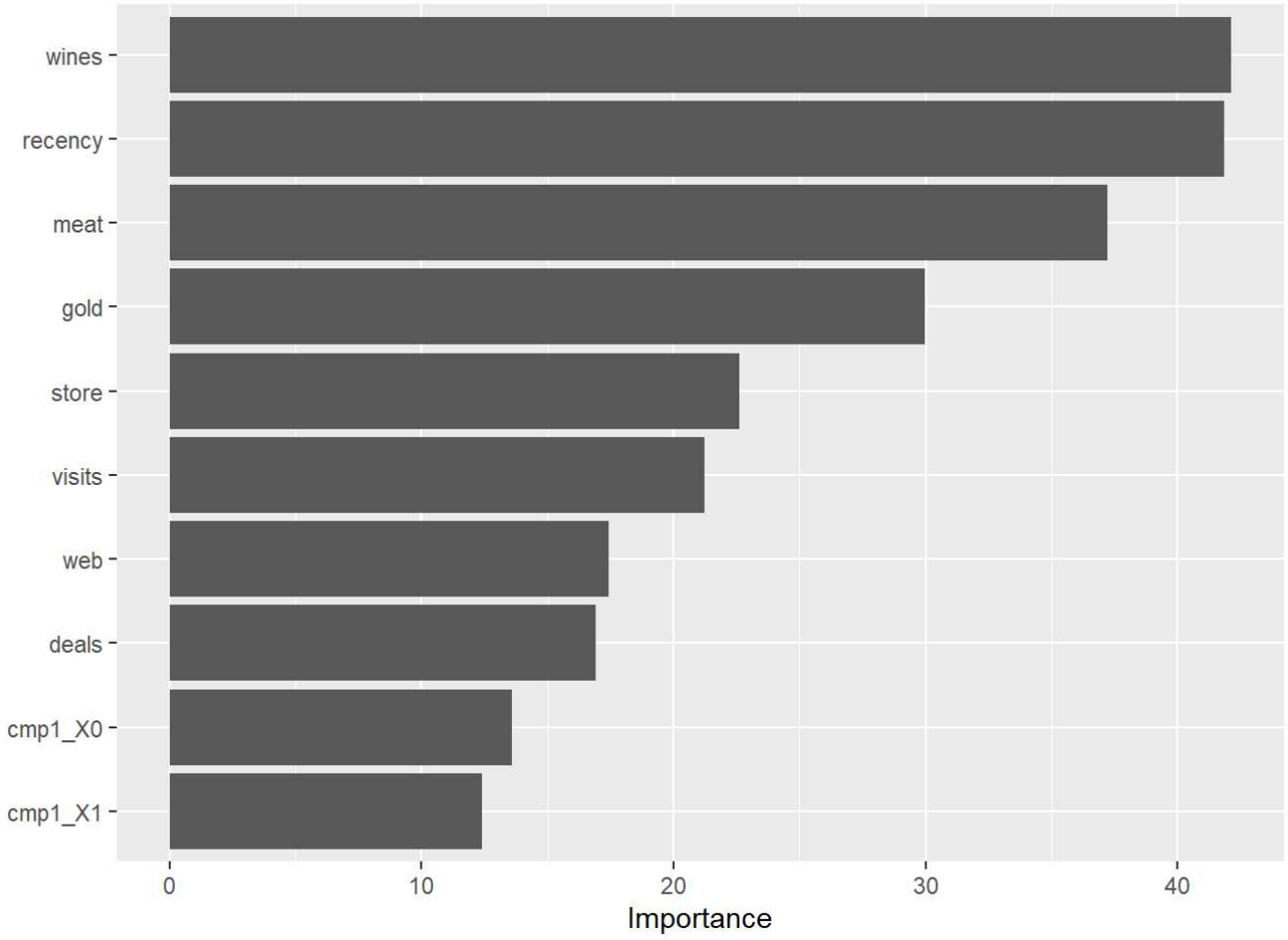
```
# Variable Importance
rf_workflow %>%
  extract_fit_parsnip() %>%
  vi()
```

Variable	Importance
<chr>	<dbl>
wines	42.119682
recency	41.850876

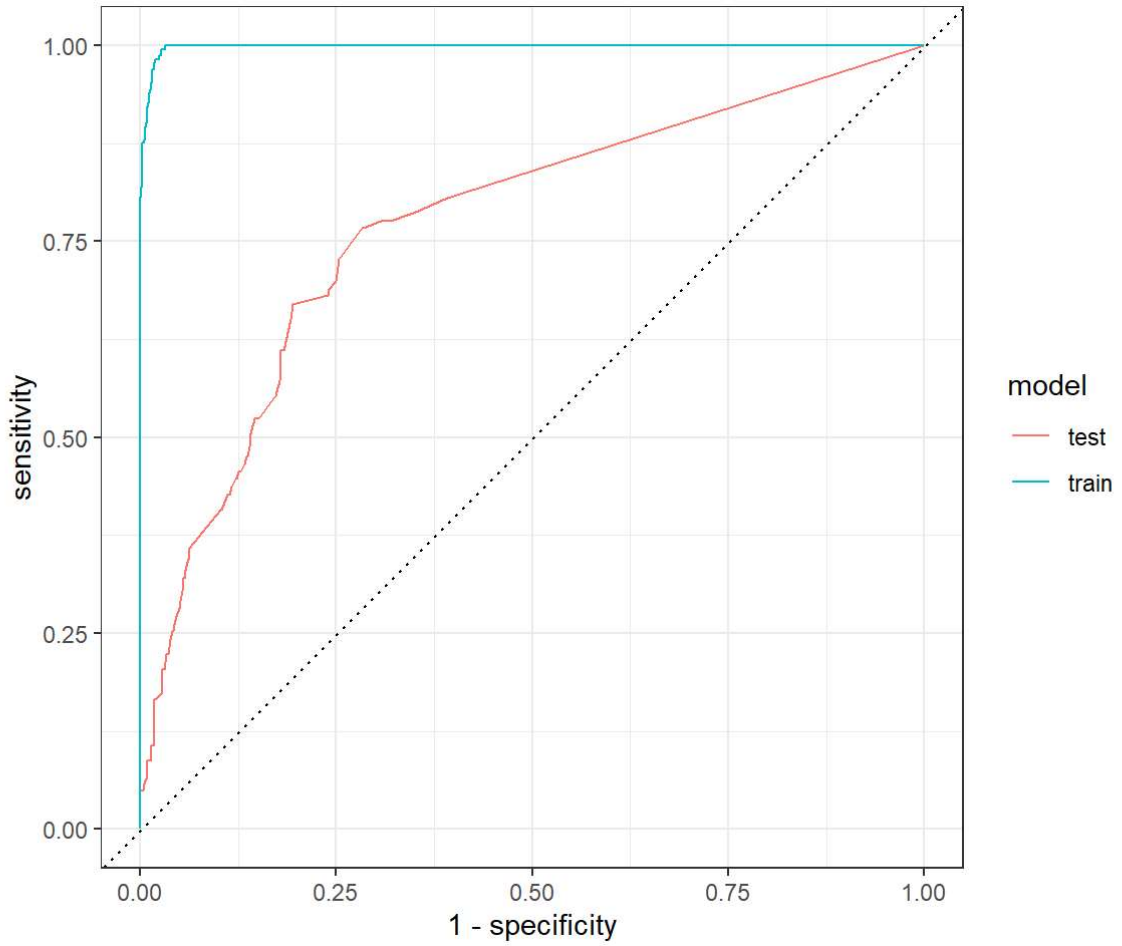
Variable	Importance
<chr>	<dbl>
meat	37.239091
gold	29.973373
store	22.613222
visits	21.211805
web	17.432655
deals	16.895481
cmp1_X0	13.599735
cmp1_X1	12.385909

1-10 of 19 rows Previous **1** 2 Next

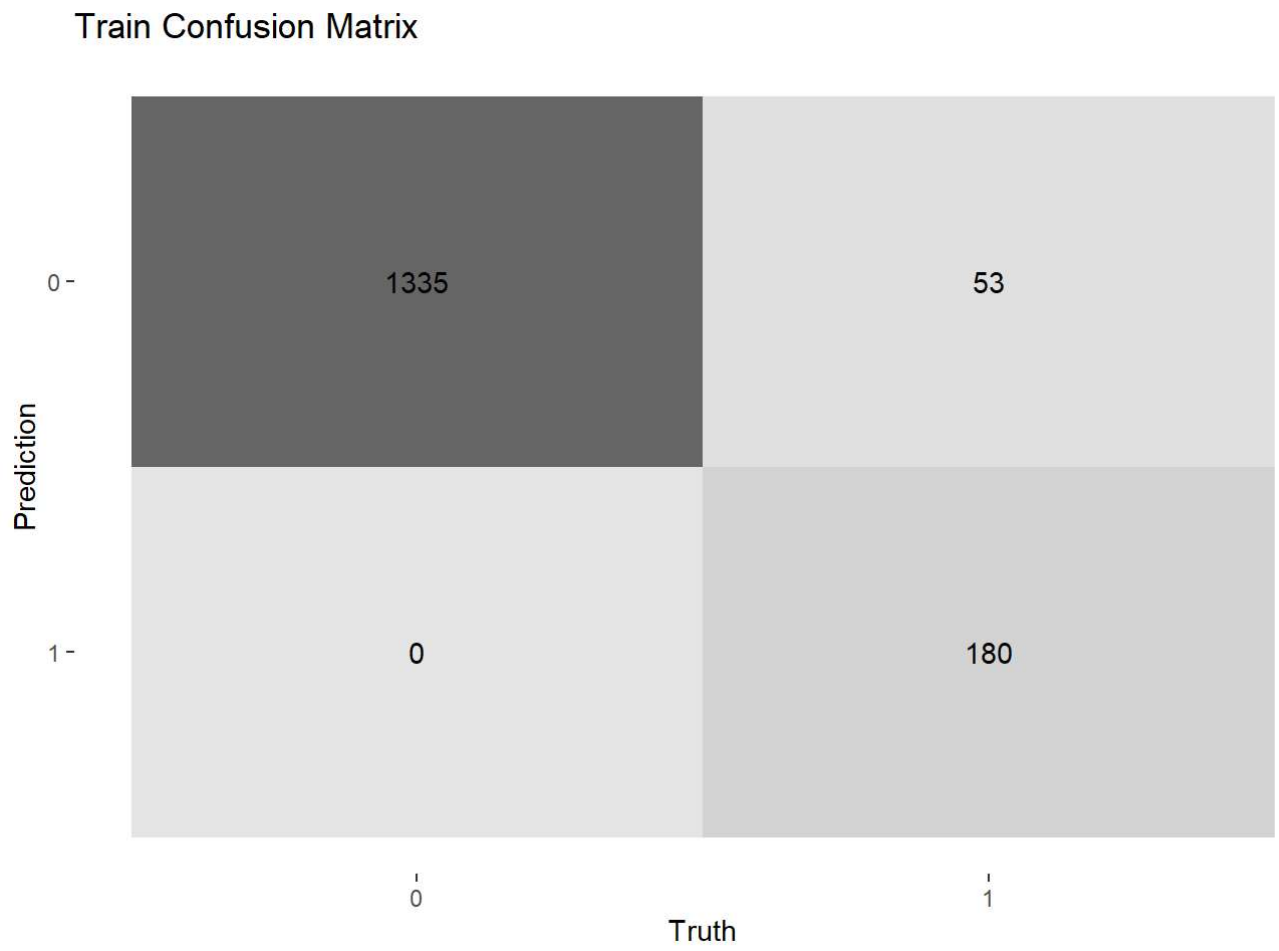
```
rf_workflow %>%
  extract_fit_parsnip() %>%
  vip()
```



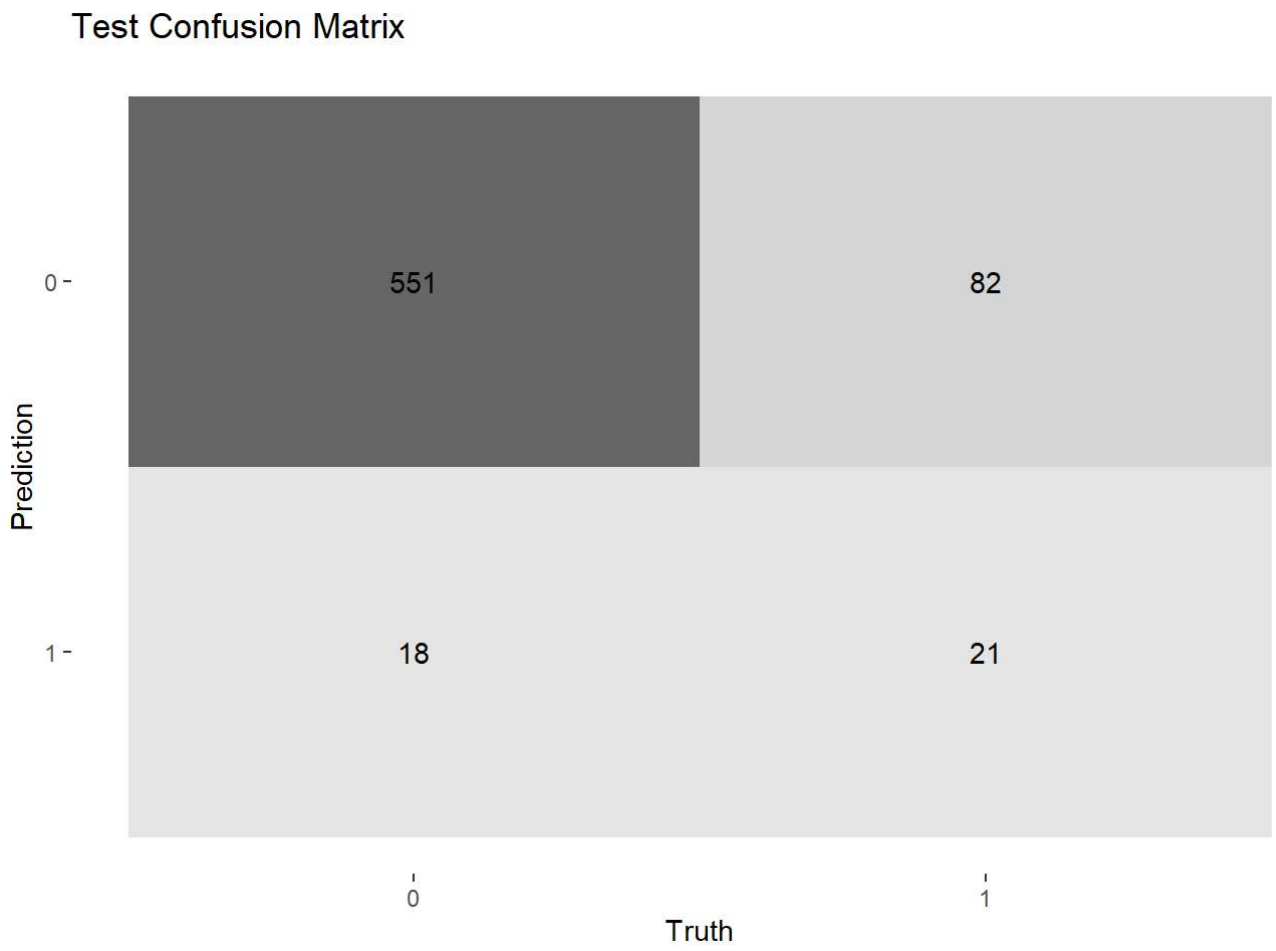
```
# ROC Charts
scored_train_rf %>%
  mutate(model = "train") %>%
  bind_rows(scored_test_knn %>%
              mutate(model="test")) %>%
  group_by(model) %>%
  roc_curve(response, .pred_1) %>%
  autoplot()
```




```
scored_train_rf %>%
  conf_mat(response, .pred_class) %>%
  autoplot(type = "heatmap") +
  labs(title="Train Confusion Matrix")
```




```
scored_test_rf %>%
  conf_mat(response, .pred_class) %>%
  autoplot(type = "heatmap") +
  labs(title="Test Confusion Matrix")
```



Random Forest

```
kfold_splits <- vfold_cv(train, v=5)

rf_model <- rand_forest(trees=tune()) %>%
  set_engine("ranger", num.threads = 5, max.depth = 10, importance="permutation") %>%
  set_mode("classification")

rf_wflow <- workflow() %>%
  add_recipe(recipe) %>%
  add_model(rf_model)

rf_search_res <- rf_wflow %>%
  tune_bayes(
    resamples = kfold_splits,
    # Generate five at semi-random to start
    initial = 5,
    iter = 50,
    metrics = metric_set(yardstick::accuracy, yardstick::roc_auc),
    control = control_bayes(no_improve = 5, verbose = TRUE)
  )
```

```
##
```

```
## ❯ Generating a set of 5 initial parameter results
```

```
## ✓ Initialization complete
```

```
##
```

```
##
```

```
## ── Iteration 1 ──────────────────────────────────────────────────────────────────────────────────────────────────────
```

```
##
```

```
## i Current best:      accuracy=0.8724 (@iter 0)
```

```
## i Gaussian process model
```

```
## ✓ Gaussian process model
```

```
## i Generating 1995 candidates
```

```
## i Predicted candidates
```

```
## i trees=593
```

```
## i Estimating performance
```

```
## i Fold1: preprocessor 1/1
```

```
## ✓ Fold1: preprocessor 1/1
```

```
## i Fold1: preprocessor 1/1, model 1/1
```

```
## ✓ Fold1: preprocessor 1/1, model 1/1
```

```
## i Fold1: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold2: preprocessor 1/1
```

```
## ✓ Fold2: preprocessor 1/1
```

```
## i Fold2: preprocessor 1/1, model 1/1
```

```
## ✓ Fold2: preprocessor 1/1, model 1/1
```

```
## i Fold2: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold3: preprocessor 1/1
```

```
## ✓ Fold3: preprocessor 1/1
```

```
## i Fold3: preprocessor 1/1, model 1/1
```

```
## ✓ Fold3: preprocessor 1/1, model 1/1
```

```
## i Fold3: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold4: preprocessor 1/1
```

```
## ✓ Fold4: preprocessor 1/1
```

```
## i Fold4: preprocessor 1/1, model 1/1
```

```
## ✓ Fold4: preprocessor 1/1, model 1/1
```

```
## i Fold4: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold5: preprocessor 1/1
```

```
## ✓ Fold5: preprocessor 1/1
```

```
## i Fold5: preprocessor 1/1, model 1/1
```

```
## ✓ Fold5: preprocessor 1/1, model 1/1
```

```
## i Fold5: preprocessor 1/1, model 1/1 (predictions)
```

```
## ✓ Estimating performance
```

```
## ⓧ Newest results:    accuracy=0.868 (+/-0.00749)
```

```
##
```

```
## ——— Iteration 2 ———————————————————————————————————————————————————————————————
```

```
##
```

```
## i Current best:      accuracy=0.8724 (@iter 0)
```

```
## i Gaussian process model
```

```
## ✓ Gaussian process model
```

```
## i Generating 1994 candidates
```

```
## i Predicted candidates
```

```
## i trees=431
```

```
## i Estimating performance
```

```
## i Fold1: preprocessor 1/1
```

```
## ✓ Fold1: preprocessor 1/1
```

```
## i Fold1: preprocessor 1/1, model 1/1
```

```
## ✓ Fold1: preprocessor 1/1, model 1/1
```

```
## i Fold1: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold2: preprocessor 1/1
```

```
## ✓ Fold2: preprocessor 1/1
```

```
## i Fold2: preprocessor 1/1, model 1/1
```

```
## ✓ Fold2: preprocessor 1/1, model 1/1
```

```
## i Fold2: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold3: preprocessor 1/1
```

```
## ✓ Fold3: preprocessor 1/1
```

```
## i Fold3: preprocessor 1/1, model 1/1
```

```
## ✓ Fold3: preprocessor 1/1, model 1/1
```

```
## i Fold3: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold4: preprocessor 1/1
```

```
## ✓ Fold4: preprocessor 1/1
```

```
## i Fold4: preprocessor 1/1, model 1/1
```

```
## ✓ Fold4: preprocessor 1/1, model 1/1
```

```
## i Fold4: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold5: preprocessor 1/1
```

```
## ✓ Fold5: preprocessor 1/1
```

```
## i Fold5: preprocessor 1/1, model 1/1
```

```
## ✓ Fold5: preprocessor 1/1, model 1/1
```

```
## i Fold5: preprocessor 1/1, model 1/1 (predictions)
```

```
## ✓ Estimating performance
```

```
## ⓧ Newest results:    accuracy=0.8686 (+/-0.00868)
```

```
##
```

```
## ── Iteration 3 ──────────────────────────────────────────────────────────────
──────────────────────────────────
```

```
##
```

```
## i Current best:      accuracy=0.8724 (@iter 0)
```

```
## i Gaussian process model
```

```
## ✓ Gaussian process model
```

```
## i Generating 1993 candidates
```

```
## i Predicted candidates
```

```
## i trees=1555
```

```
## i Estimating performance
```

```
## i Fold1: preprocessor 1/1
```

```
## ✓ Fold1: preprocessor 1/1
```

```
## i Fold1: preprocessor 1/1, model 1/1
```

```
## ✓ Fold1: preprocessor 1/1, model 1/1
```

```
## i Fold1: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold2: preprocessor 1/1
```

```
## ✓ Fold2: preprocessor 1/1
```

```
## i Fold2: preprocessor 1/1, model 1/1
```

```
## ✓ Fold2: preprocessor 1/1, model 1/1
```

```
## i Fold2: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold3: preprocessor 1/1
```

```
## ✓ Fold3: preprocessor 1/1
```

```
## i Fold3: preprocessor 1/1, model 1/1
```

```
## ✓ Fold3: preprocessor 1/1, model 1/1
```

```
## i Fold3: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold4: preprocessor 1/1
```

```
## ✓ Fold4: preprocessor 1/1
```

```
## i Fold4: preprocessor 1/1, model 1/1
```

```
## ✓ Fold4: preprocessor 1/1, model 1/1
```

```
## i Fold4: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold5: preprocessor 1/1
```

```
## ✓ Fold5: preprocessor 1/1
```

```
## i Fold5: preprocessor 1/1, model 1/1
```

```
## ✓ Fold5: preprocessor 1/1, model 1/1
```

```
## i Fold5: preprocessor 1/1, model 1/1 (predictions)
```

```
## ✓ Estimating performance
```

```
## ⓧ Newest results:    accuracy=0.868 (+/-0.00876)
```

```
##
```

```
## ── Iteration 4 ──────────────────────────────────────────────────────────────
──────────────────────────────
```

```
##
```

```
## i Current best:      accuracy=0.8724 (@iter 0)
```

```
## i Gaussian process model
```

```
## ✓ Gaussian process model
```

```
## i Generating 1992 candidates
```

```
## i Predicted candidates
```

```
## i trees=220
```

```
## i Estimating performance
```

```
## i Fold1: preprocessor 1/1
```

```
## ✓ Fold1: preprocessor 1/1
```

```
## i Fold1: preprocessor 1/1, model 1/1
```

```
## ✓ Fold1: preprocessor 1/1, model 1/1
```

```
## i Fold1: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold2: preprocessor 1/1
```

```
## ✓ Fold2: preprocessor 1/1
```

```
## i Fold2: preprocessor 1/1, model 1/1
```

```
## ✓ Fold2: preprocessor 1/1, model 1/1
```

```
## i Fold2: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold3: preprocessor 1/1
```

```
## ✓ Fold3: preprocessor 1/1
```

```
## i Fold3: preprocessor 1/1, model 1/1
```

```
## ✓ Fold3: preprocessor 1/1, model 1/1
```

```
## i Fold3: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold4: preprocessor 1/1
```

```
## ✓ Fold4: preprocessor 1/1
```

```
## i Fold4: preprocessor 1/1, model 1/1
```

```
## ✓ Fold4: preprocessor 1/1, model 1/1
```

```
## i Fold4: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold5: preprocessor 1/1
```

```
## ✓ Fold5: preprocessor 1/1
```

```
## i Fold5: preprocessor 1/1, model 1/1
```

```
## ✓ Fold5: preprocessor 1/1, model 1/1
```

```
## i Fold5: preprocessor 1/1, model 1/1 (predictions)
```

```
## ✓ Estimating performance
```

```
## ⓧ Newest results:    accuracy=0.8711 (+/-0.00887)
```

```
##
```

```
## ─── Iteration 5 ──────────────────────────────────────────────────────────────
──────────────────────────────────────
```

```
##
```

```
## i Current best:      accuracy=0.8724 (@iter 0)
```

```
## i Gaussian process model
```

```
## ✓ Gaussian process model
```

```
## i Generating 1991 candidates
```

```
## i Predicted candidates
```

```
## i trees=1148
```

```
## i Estimating performance
```

```
## i Fold1: preprocessor 1/1
```

```
## ✓ Fold1: preprocessor 1/1
```

```
## i Fold1: preprocessor 1/1, model 1/1
```

```
## ✓ Fold1: preprocessor 1/1, model 1/1
```

```
## i Fold1: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold2: preprocessor 1/1
```

```
## ✓ Fold2: preprocessor 1/1
```

```
## i Fold2: preprocessor 1/1, model 1/1
```

```
## ✓ Fold2: preprocessor 1/1, model 1/1
```

```
## i Fold2: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold3: preprocessor 1/1
```

```
## ✓ Fold3: preprocessor 1/1
```

```
## i Fold3: preprocessor 1/1, model 1/1
```

```
## ✓ Fold3: preprocessor 1/1, model 1/1
```

```
## i Fold3: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold4: preprocessor 1/1
```

```
## ✓ Fold4: preprocessor 1/1
```

```
## i Fold4: preprocessor 1/1, model 1/1
```

```
## ✓ Fold4: preprocessor 1/1, model 1/1
```

```
## i Fold4: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold5: preprocessor 1/1
```

```
## ✓ Fold5: preprocessor 1/1
```

```
## i Fold5: preprocessor 1/1, model 1/1
```

```
## ✓ Fold5: preprocessor 1/1, model 1/1
```

```
## i Fold5: preprocessor 1/1, model 1/1 (predictions)
```

```
## ✓ Estimating performance
```

```
## ⓧ Newest results:    accuracy=0.8712 (+/-0.00749)
```

```
## ! No improvement for 5 iterations; returning current results.
```

Final Fit Random Forest

```
highest_rf_accuracy <- rf_search_res %>%
  select_best("accuracy")

highest_rf_accuracy
```

trees	.config
<int>	<chr>
445	Preprocessor1_Model5

1 row

```
rf_wflow <- finalize_workflow(
  rf_wflow, highest_rf_accuracy
) %>%
  fit(train)
```

Evaluate the Random Forest Model

```
options(yardstick.event_first = FALSE)
  # score training
  predict(rf_wflow, train, type="prob") %>%
    bind_cols(predict(rf_wflow, train, type="class")) %>%
    bind_cols(., train)-> scored_train_rf_tune

  # score testing
  predict(rf_wflow, test, type="prob") %>%
      bind_cols(predict(rf_wflow, test, type="class")) %>%
      bind_cols(., test) -> scored_test_rf_tune

# Metrics: Train and Test
scored_train_rf_tune %>%
    metrics(response, .pred_1, estimate = .pred_class) %>%
    mutate(part="training") %>%
    bind_rows(scored_test_rf_tune %>%
                metrics(response, .pred_1, estimate = .pred_class) %>%
                mutate(part="testing") ) %>%
    filter(.metric %in% c('accuracy','roc_auc')) %>%
    pivot_wider(names_from = .metric, values_from=.estimate)
```

.estimator	part	accuracy	roc_auc
<chr>	<chr>	<dbl>	<dbl>
binary	training	0.9470663	0.9920963
binary	testing	0.8556548	0.8349259

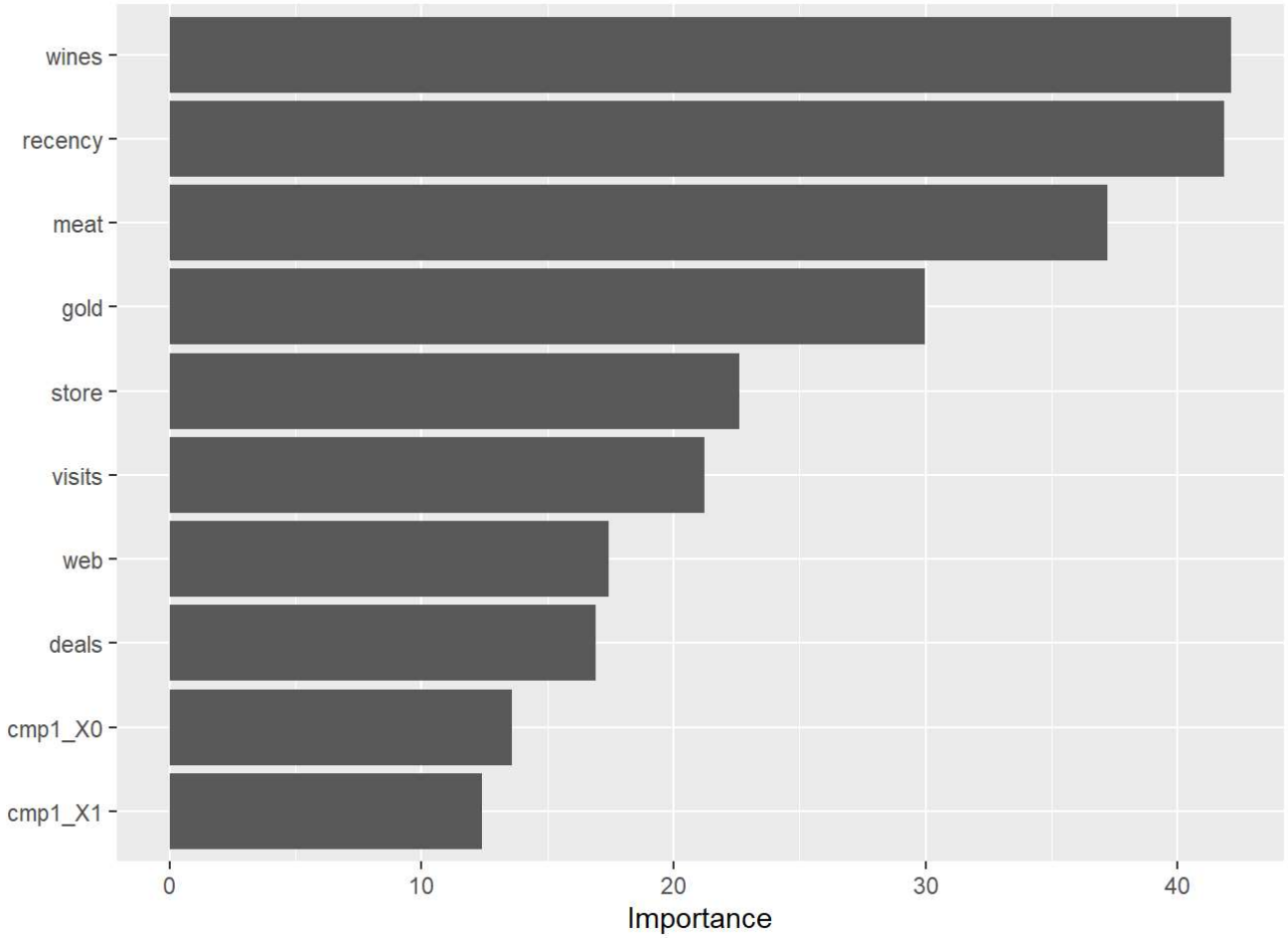
2 rows

```
# Variable Importance
rf_workflow %>%
  extract_fit_parsnip() %>%
  vi()
```

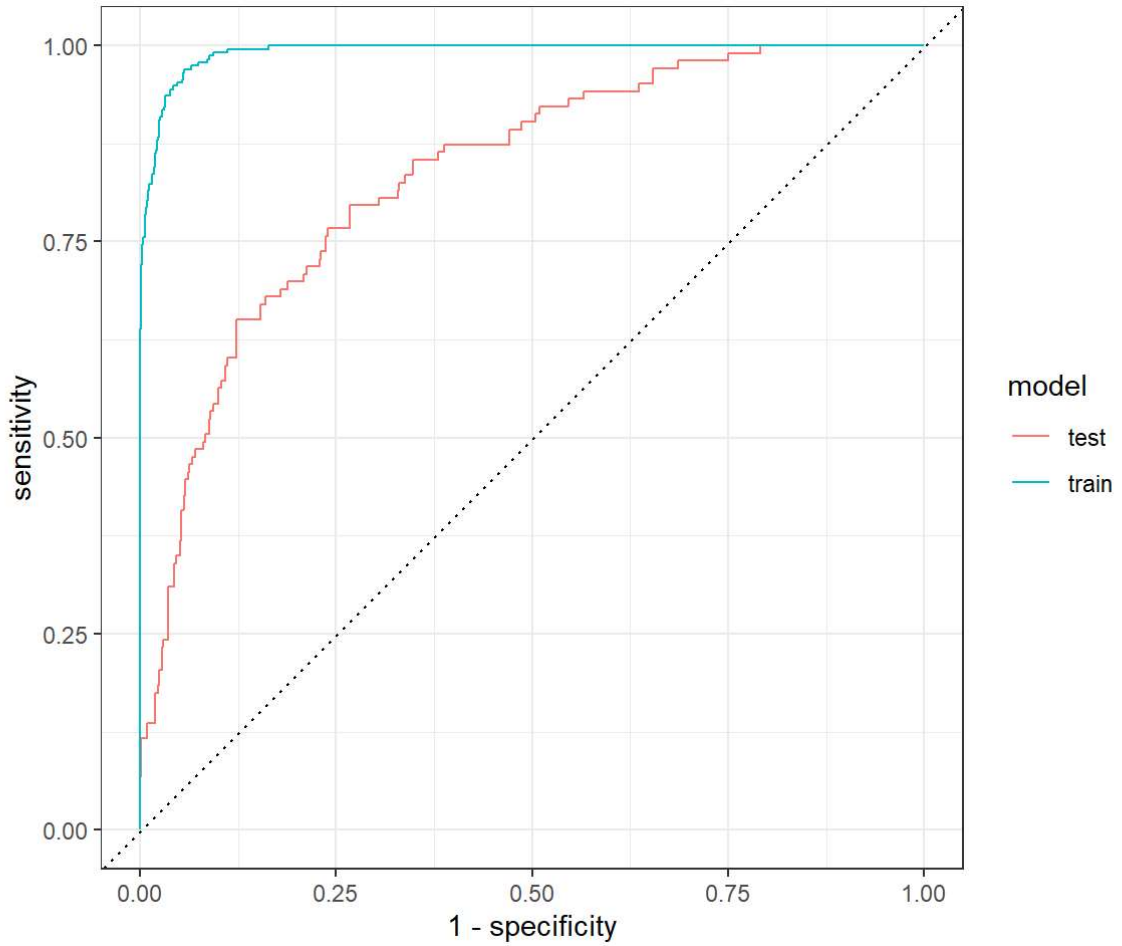
Variable	Importance
<chr>	<dbl>
wines	42.119682
recency	41.850876
meat	37.239091
gold	29.973373
store	22.613222
visits	21.211805
web	17.432655
deals	16.895481
cmp1_X0	13.599735
cmp1_X1	12.385909

1-10 of 19 rows Previous **1** 2 Next

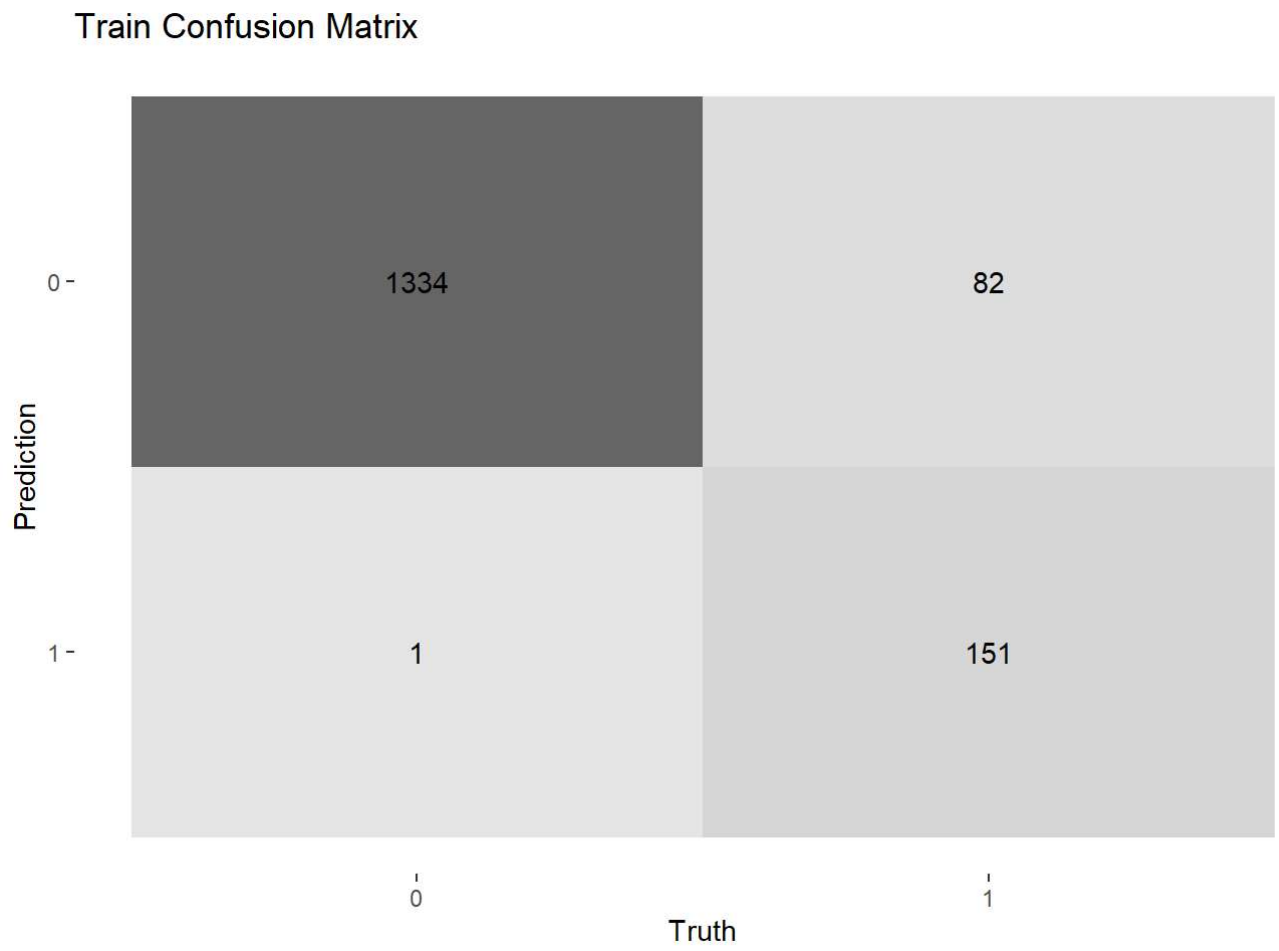
```
rf_workflow %>%
  extract_fit_parsnip() %>%
  vip()
```



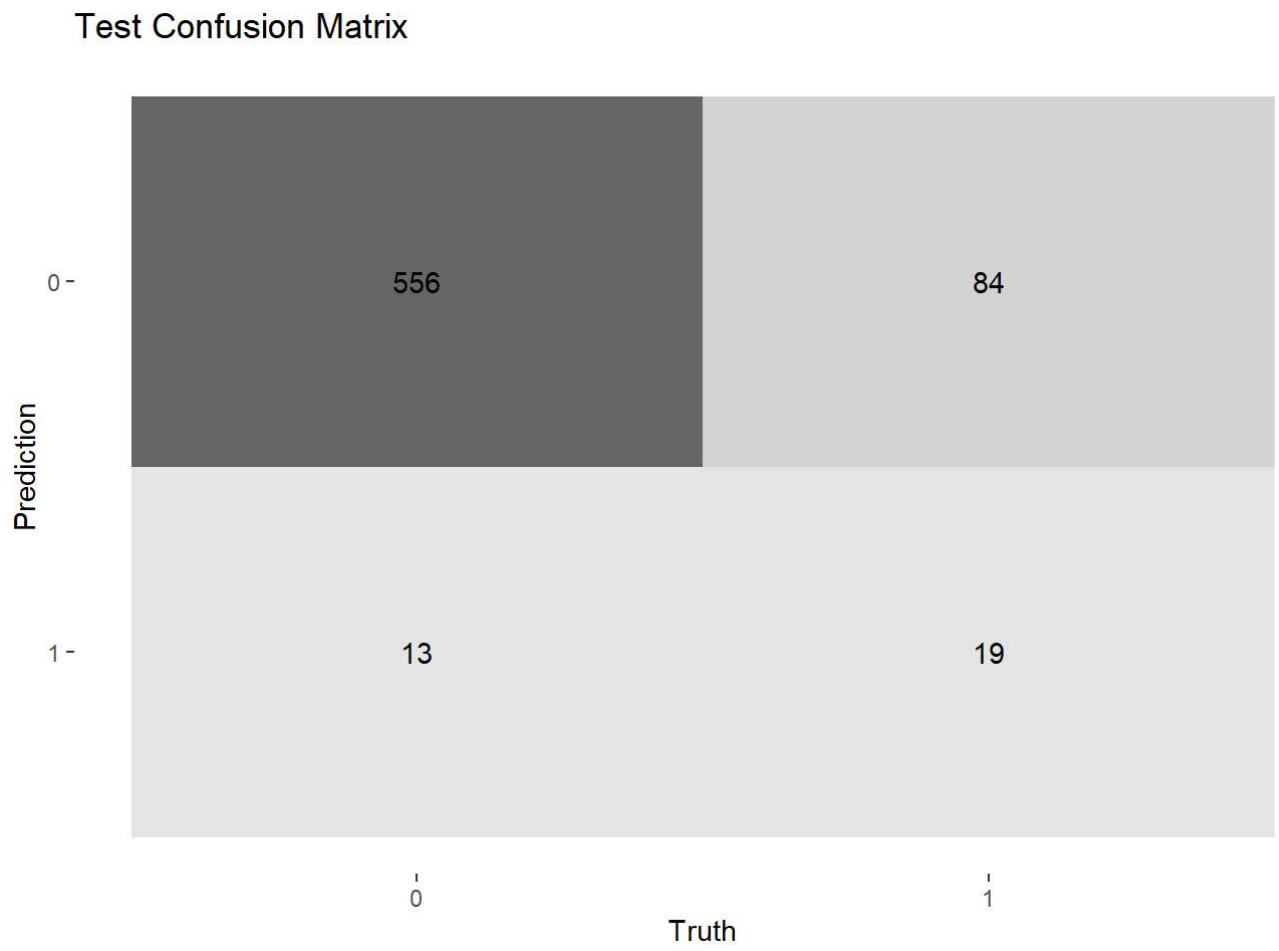

```
# ROC Charts
scored_train_rf_tune %>%
  mutate(model = "train") %>%
  bind_rows(scored_test_rf_tune %>%
              mutate(model="test")) %>%
  group_by(model) %>%
  roc_curve(response, .pred_1) %>%
  autoplot()
```




```
scored_train_rf_tune %>%
  conf_mat(response, .pred_class) %>%
  autoplot(type = "heatmap") +
  labs(title="Train Confusion Matrix")
```




```
scored_test_rf_tune %>%
  conf_mat(response, .pred_class) %>%
  autoplot(type = "heatmap") +
  labs(title="Test Confusion Matrix")
```



XGBoost Model Buiding

Here we want to TUNE our XGB model using the Bayes method.

```
xgb_model <- boost_tree(trees = tune(),
                        learn_rate = tune(),
                        tree_depth = tune()) %>%
  set_engine("xgboost",
             importance="permutation") %>%
  set_mode("classification")

xgb_wflow <- workflow() %>%
  add_recipe(recipe) %>%
  add_model(xgb_model)

xgb_search_res <- xgb_wflow %>%
  tune_bayes(
    resamples = kfold_splits,
    # Generate five at semi-random to start
    initial = 5,
    iter = 50,
    metrics = metric_set(yardstick::accuracy, yardstick::roc_auc),
    control = control_bayes(no_improve = 5, verbose = TRUE)
  )
```

```
##
```

```
## ❯  Generating a set of 5 initial parameter results
```

```
## ✓ Initialization complete
```

```
##
```

```
##
```

```
## ──── Iteration 1 ──────────────────────────────────────────────────────────────────────────────
```

```
##
```

```
## i Current best:      accuracy=0.8833 (@iter 0)
```

```
## i Gaussian process model
```

```
## ✓ Gaussian process model
```

```
## i Generating 5000 candidates
```

```
## i Predicted candidates
```

```
## i trees=1172, tree_depth=6, learn_rate=0.316
```

```
## i Estimating performance
```

```
## i Fold1: preprocessor 1/1
```

```
## ✓ Fold1: preprocessor 1/1
```

```
## i Fold1: preprocessor 1/1, model 1/1
```

```
## ✓ Fold1: preprocessor 1/1, model 1/1
```

```
## i Fold1: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold2: preprocessor 1/1
```

```
## ✓ Fold2: preprocessor 1/1
```

```
## i Fold2: preprocessor 1/1, model 1/1
```

```
## ✓ Fold2: preprocessor 1/1, model 1/1
```

```
## i Fold2: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold3: preprocessor 1/1
```

```
## ✓ Fold3: preprocessor 1/1
```

```
## i Fold3: preprocessor 1/1, model 1/1
```

```
## ✓ Fold3: preprocessor 1/1, model 1/1
```

```
## i Fold3: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold4: preprocessor 1/1
```

```
## ✓ Fold4: preprocessor 1/1
```

```
## i Fold4: preprocessor 1/1, model 1/1
```

```
## ✓ Fold4: preprocessor 1/1, model 1/1
```

```
## i Fold4: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold5: preprocessor 1/1
```

```
## ✓ Fold5: preprocessor 1/1
```

```
## i Fold5: preprocessor 1/1, model 1/1
```

```
## ✓ Fold5: preprocessor 1/1, model 1/1
```

```
## i Fold5: preprocessor 1/1, model 1/1 (predictions)
```

```
## ✓ Estimating performance
```

```
## ⓧ Newest results:    accuracy=0.8769 (+/-0.0089)
```

```
##
```

```
## ── Iteration 2 ──────────────────────────────────────────────────────────────────
──────────────────────────────────
```

```
##
```

```
## i Current best:      accuracy=0.8833 (@iter 0)
```

```
## i Gaussian process model
```

```
## ✓ Gaussian process model
```

```
## i Generating 5000 candidates
```

```
## i Predicted candidates
```

```
## i trees=270, tree_depth=4, learn_rate=0.0442
```

```
## i Estimating performance
```

```
## i Fold1: preprocessor 1/1
```

```
## ✓ Fold1: preprocessor 1/1
```

```
## i Fold1: preprocessor 1/1, model 1/1
```

```
## ✓ Fold1: preprocessor 1/1, model 1/1
```

```
## i Fold1: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold2: preprocessor 1/1
```

```
## ✓ Fold2: preprocessor 1/1
```

```
## i Fold2: preprocessor 1/1, model 1/1
```

```
## ✓ Fold2: preprocessor 1/1, model 1/1
```

```
## i Fold2: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold3: preprocessor 1/1
```

```
## ✓ Fold3: preprocessor 1/1
```

```
## i Fold3: preprocessor 1/1, model 1/1
```

```
## ✓ Fold3: preprocessor 1/1, model 1/1
```

```
## i Fold3: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold4: preprocessor 1/1
```

```
## ✓ Fold4: preprocessor 1/1
```

```
## i Fold4: preprocessor 1/1, model 1/1
```

```
## ✓ Fold4: preprocessor 1/1, model 1/1
```

```
## i Fold4: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold5: preprocessor 1/1
```

```
## ✓ Fold5: preprocessor 1/1
```

```
## i Fold5: preprocessor 1/1, model 1/1
```

```
## ✓ Fold5: preprocessor 1/1, model 1/1
```

```
## i Fold5: preprocessor 1/1, model 1/1 (predictions)
```

```
## ✓ Estimating performance
```

```
## ⓧ Newest results:    accuracy=0.8788 (+/-0.00646)
```

```
##
```

```
## ── Iteration 3 ──────────────────────────────────────────────────────────────
──────────────────────────────────────
```

```
##
```

```
## i Current best:      accuracy=0.8833 (@iter 0)
```

```
## i Gaussian process model
```

```
## ✓ Gaussian process model
```

```
## i Generating 5000 candidates
```

```
## i Predicted candidates
```

```
## i trees=266, tree_depth=1, learn_rate=0.0103
```

```
## i Estimating performance
```

```
## i Fold1: preprocessor 1/1
```

```
## ✓ Fold1: preprocessor 1/1
```

```
## i Fold1: preprocessor 1/1, model 1/1
```

```
## ✓ Fold1: preprocessor 1/1, model 1/1
```

```
## i Fold1: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold2: preprocessor 1/1
```

```
## ✓ Fold2: preprocessor 1/1
```

```
## i Fold2: preprocessor 1/1, model 1/1
```

```
## ✓ Fold2: preprocessor 1/1, model 1/1
```

```
## i Fold2: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold3: preprocessor 1/1
```

```
## ✓ Fold3: preprocessor 1/1
```

```
## i Fold3: preprocessor 1/1, model 1/1
```

```
## ✓ Fold3: preprocessor 1/1, model 1/1
```

```
## i Fold3: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold4: preprocessor 1/1
```

```
## ✓ Fold4: preprocessor 1/1
```

```
## i Fold4: preprocessor 1/1, model 1/1
```

```
## ✓ Fold4: preprocessor 1/1, model 1/1
```

```
## i Fold4: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold5: preprocessor 1/1
```

```
## ✓ Fold5: preprocessor 1/1
```

```
## i Fold5: preprocessor 1/1, model 1/1
```

```
## ✓ Fold5: preprocessor 1/1, model 1/1
```

```
## i Fold5: preprocessor 1/1, model 1/1 (predictions)
```

```
## ✓ Estimating performance
```

```
## ⓧ Newest results:    accuracy=0.852 (+/-0.00706)
```

```
##
```

```
## ── Iteration 4 ──────────────────────────────────────────────────────────────
```

```
##
```

```
## i Current best:      accuracy=0.8833 (@iter 0)
```

```
## i Gaussian process model
```

```
## ✓ Gaussian process model
```

```
## i Generating 5000 candidates
```

```
## i Predicted candidates
```

```
## i trees=1991, tree_depth=2, learn_rate=0.0758
```

```
## i Estimating performance
```

```
## i Fold1: preprocessor 1/1
```

```
## ✓ Fold1: preprocessor 1/1
```

```
## i Fold1: preprocessor 1/1, model 1/1
```

```
## ✓ Fold1: preprocessor 1/1, model 1/1
```

```
## i Fold1: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold2: preprocessor 1/1
```

```
## ✓ Fold2: preprocessor 1/1
```

```
## i Fold2: preprocessor 1/1, model 1/1
```

```
## ✓ Fold2: preprocessor 1/1, model 1/1
```

```
## i Fold2: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold3: preprocessor 1/1
```

```
## ✓ Fold3: preprocessor 1/1
```

```
## i Fold3: preprocessor 1/1, model 1/1
```

```
## ✓ Fold3: preprocessor 1/1, model 1/1
```

```
## i Fold3: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold4: preprocessor 1/1
```

```
## ✓ Fold4: preprocessor 1/1
```

```
## i Fold4: preprocessor 1/1, model 1/1
```

```
## ✓ Fold4: preprocessor 1/1, model 1/1
```

```
## i Fold4: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold5: preprocessor 1/1
```

```
## ✓ Fold5: preprocessor 1/1
```

```
## i Fold5: preprocessor 1/1, model 1/1
```

```
## ✓ Fold5: preprocessor 1/1, model 1/1
```

```
## i Fold5: preprocessor 1/1, model 1/1 (predictions)
```

```
## ✓ Estimating performance
```

```
## ⓧ Newest results:    accuracy=0.8814 (+/-0.0076)
```

```
##
```

```
## ── Iteration 5 ──────────────────────────────────────────────────────────────
──────────────────────────────────────
```

```
##
```

```
## i Current best:      accuracy=0.8833 (@iter 0)
```

```
## i Gaussian process model
```

```
## ✓ Gaussian process model
```

```
## i Generating 5000 candidates
```

```
## i Predicted candidates
```

```
## i trees=1787, tree_depth=11, learn_rate=0.285
```

```
## i Estimating performance
```

```
## i Fold1: preprocessor 1/1
```

```
## ✓ Fold1: preprocessor 1/1
```

```
## i Fold1: preprocessor 1/1, model 1/1
```

```
## ✓ Fold1: preprocessor 1/1, model 1/1
```

```
## i Fold1: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold2: preprocessor 1/1
```

```
## ✓ Fold2: preprocessor 1/1
```

```
## i Fold2: preprocessor 1/1, model 1/1
```

```
## ✓ Fold2: preprocessor 1/1, model 1/1
```

```
## i Fold2: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold3: preprocessor 1/1
```

```
## ✓ Fold3: preprocessor 1/1
```

```
## i Fold3: preprocessor 1/1, model 1/1
```

```
## ✓ Fold3: preprocessor 1/1, model 1/1
```

```
## i Fold3: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold4: preprocessor 1/1
```

```
## ✓ Fold4: preprocessor 1/1
```

```
## i Fold4: preprocessor 1/1, model 1/1
```

```
## ✓ Fold4: preprocessor 1/1, model 1/1
```

```
## i Fold4: preprocessor 1/1, model 1/1 (predictions)
```

```
## i Fold5: preprocessor 1/1
```

```
## ✓ Fold5: preprocessor 1/1
```

```
## i Fold5: preprocessor 1/1, model 1/1
```

```
## ✓ Fold5: preprocessor 1/1, model 1/1
```

```
## i Fold5: preprocessor 1/1, model 1/1 (predictions)
```

```
## ✓ Estimating performance
```

```
## ⓧ Newest results:    accuracy=0.8782 (+/-0.00723)
```

```
## ! No improvement for 5 iterations; returning current results.
```

Final Fit XGB

```
highest_xgb_accuracy <- xgb_search_res %>%
  select_best("accuracy")

highest_xgb_accuracy
```

trees	tree_depth	learn_rate	.config
<int>	<int>	<dbl>	<chr>
263	2	0.2313404	Preprocessor1_Model1

1 row

```
xgb_wflow <- finalize_workflow(
  xgb_wflow, highest_xgb_accuracy
) %>%
  fit(train)
```

```
## [01:07:12] WARNING: amalgamation/../src/learner.cc:627:
## Parameters: { "importance" } might not be used.
##
##   This could be a false alarm, with some parameters getting used by language bindings but
##   then being mistakenly passed down to XGBoost core, or some parameter actually being used
##   but getting flagged wrongly here. Please open an issue if you find any such cases.
```

Evaluate the XGBoost Model

```
options(yardstick.event_first = FALSE)
  # score training
  predict(xgb_wflow, train, type="prob") %>%
    bind_cols(predict(xgb_wflow, train, type="class")) %>%
    bind_cols(., train) -> scored_train_xgb

  # score testing
  predict(xgb_wflow, test, type="prob") %>%
      bind_cols(predict(xgb_wflow, test, type="class")) %>%
      bind_cols(., test) -> scored_test_xgb

# Metrics: Train and Test
scored_train_xgb %>%
    metrics(response, .pred_1, estimate = .pred_class) %>%
    mutate(part="training") %>%
    bind_rows(scored_test_xgb %>%
                metrics(response, .pred_1, estimate = .pred_class) %>%
                mutate(part="testing") ) %>%
    filter(.metric %in% c('accuracy','roc_auc')) %>%
    pivot_wider(names_from = .metric, values_from=.estimate)
```

.estimator	part	accuracy	roc_auc
<chr>	<chr>	<dbl>	<dbl>
binary	training	0.9547194	0.9858851
binary	testing	0.8630952	0.8365980

2 rows

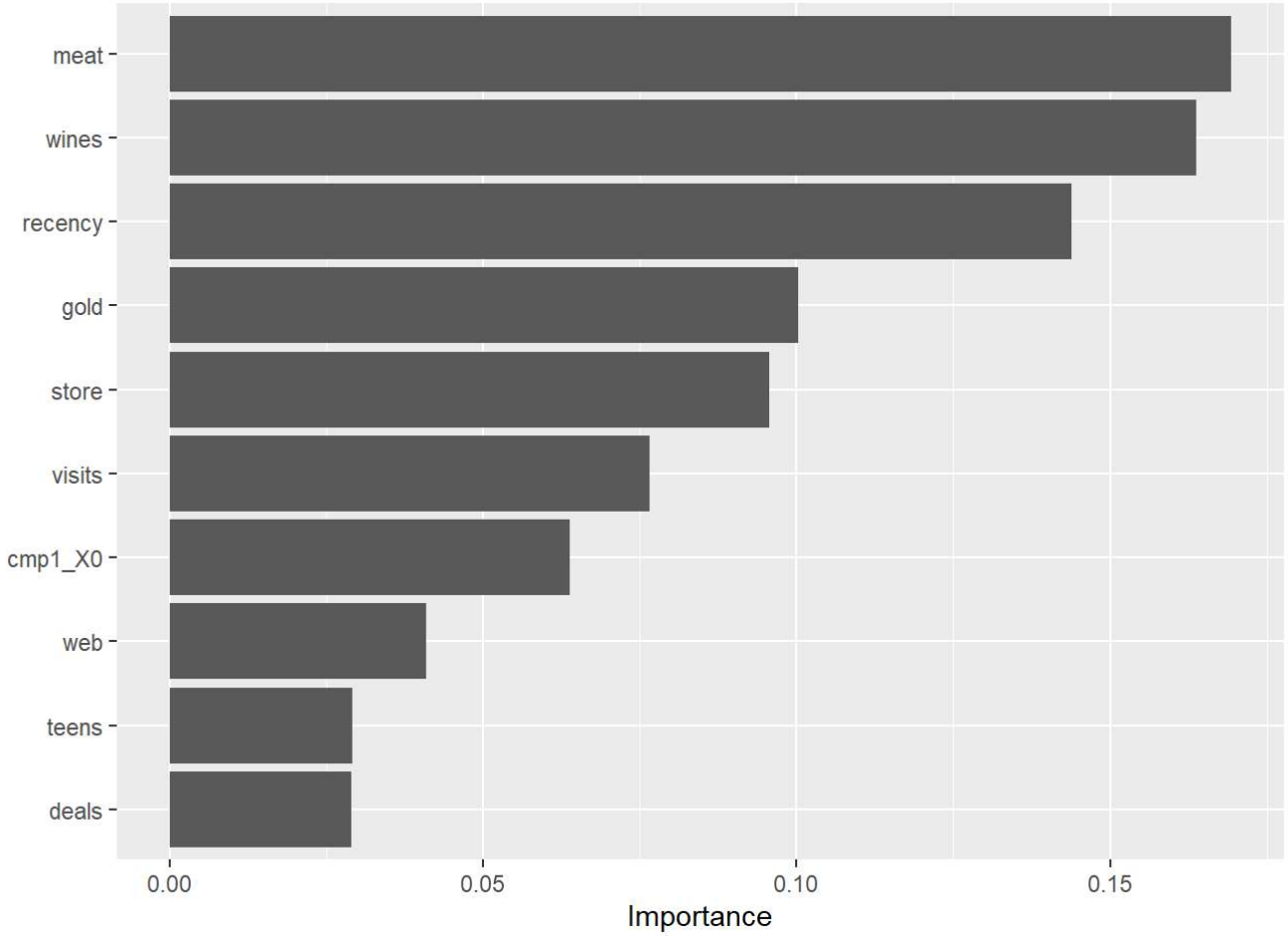
```
# Variable Importance
xgb_wflow %>%
  extract_fit_parsnip() %>%
  vi()
```

Variable	Importance
<chr>	<dbl>
meat	0.1692653287
wines	0.1637877604
recency	0.1438959418
gold	0.1003398121

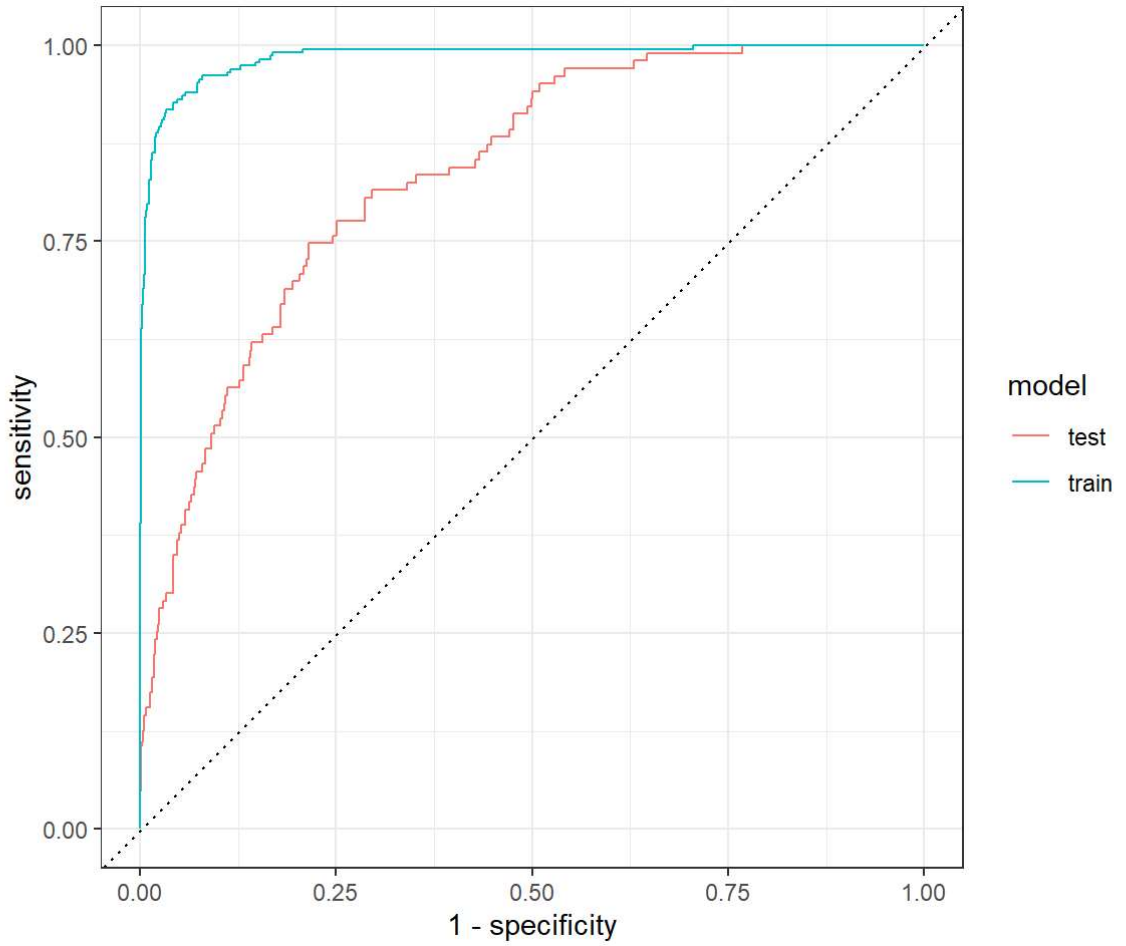
Variable	Importance
<chr>	<dbl>
store	0.0957273176
visits	0.0765613548
cmp1_X0	0.0637615020
web	0.0408948792
teens	0.0291224650
deals	0.0290649181

1-10 of 18 rows Previous **1** 2 Next

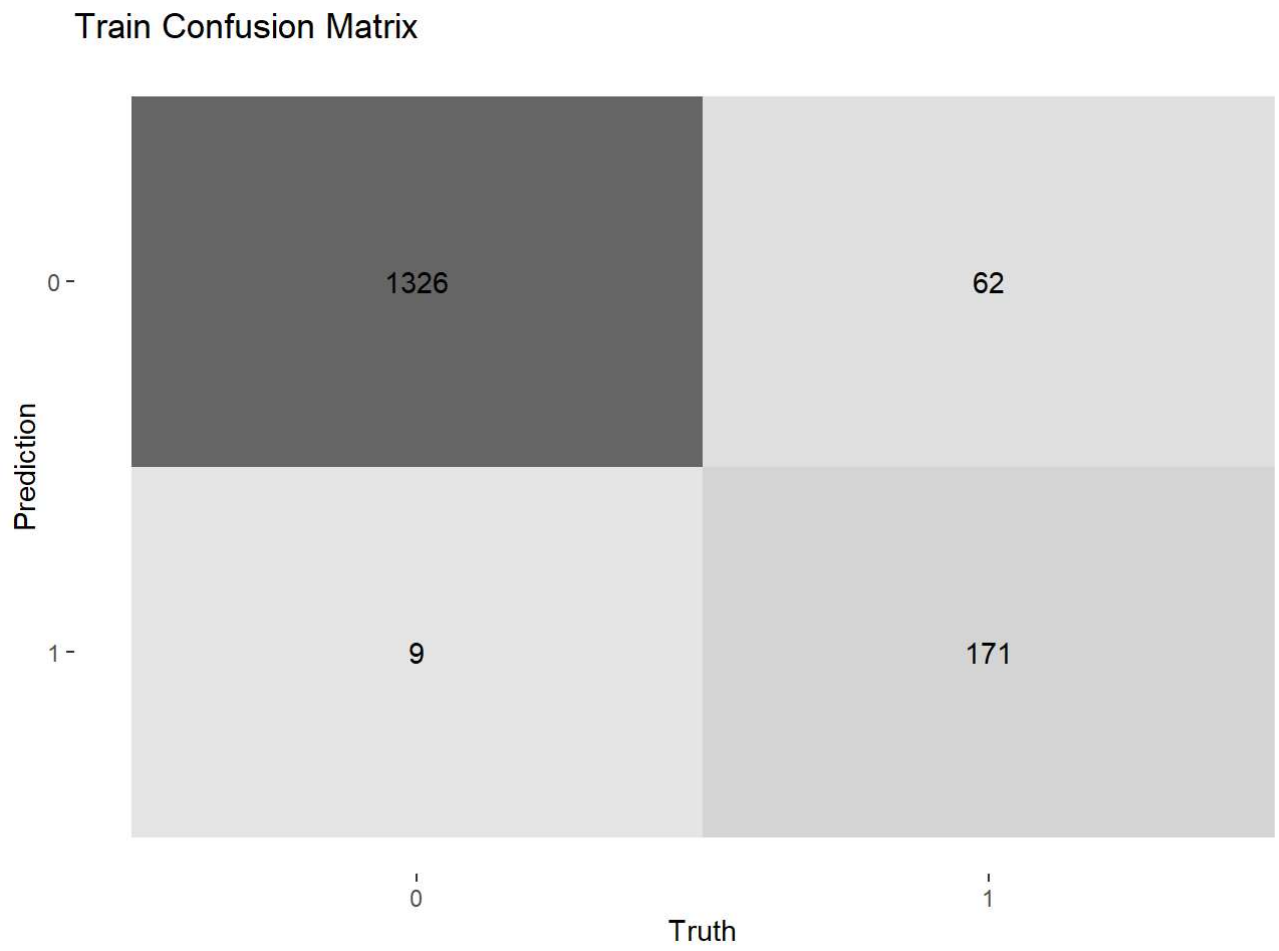
```
xgb_wflow %>%
  extract_fit_parsnip() %>%
  vip()
```



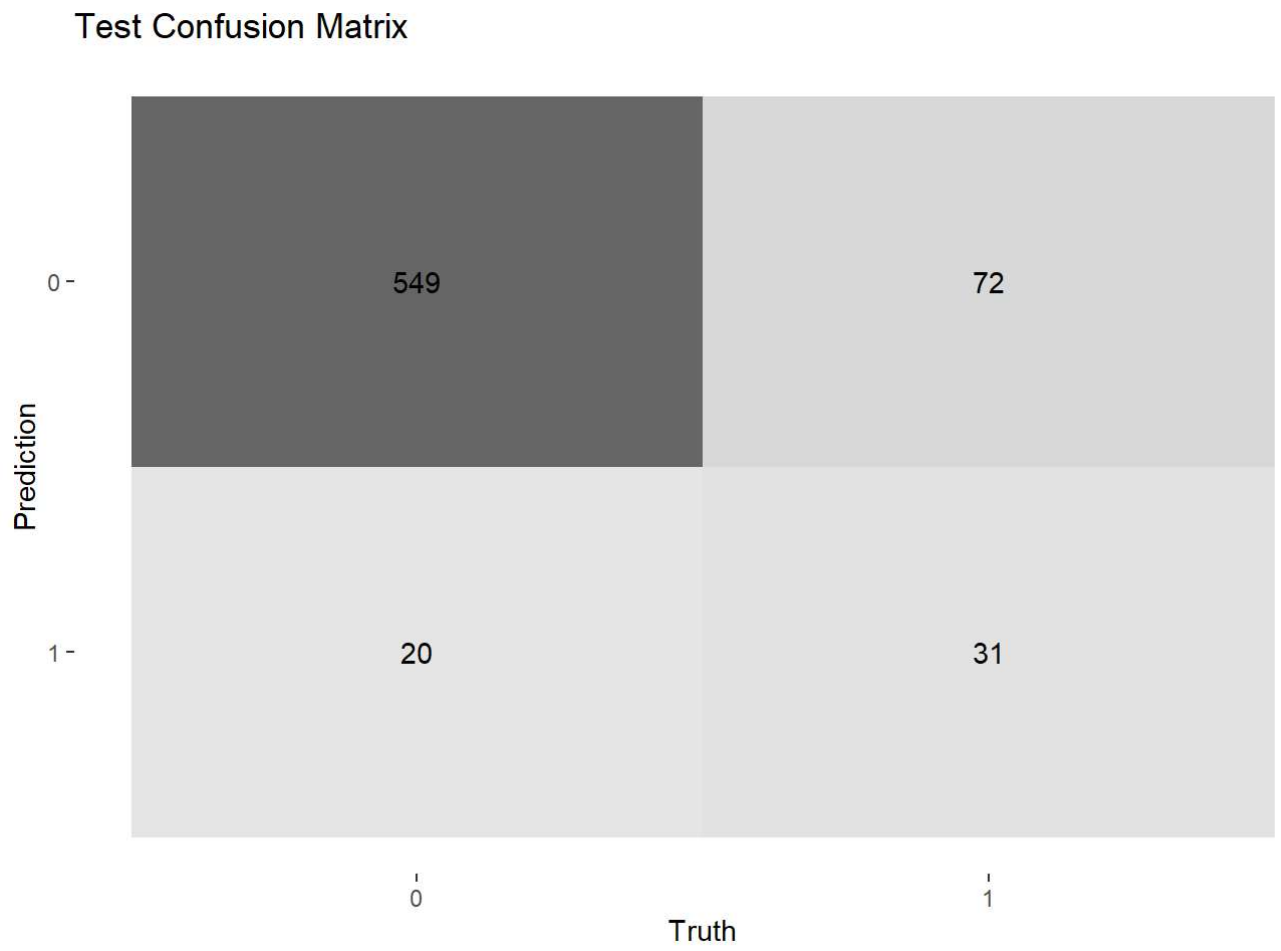
```
# ROC Charts
scored_train_xgb %>%
  mutate(model = "train") %>%
  bind_rows(scored_test_xgb %>%
              mutate(model="test")) %>%
  group_by(model) %>%
  roc_curve(response, .pred_1) %>%
  autoplot()
```




```
scored_train_xgb %>%
  conf_mat(response, .pred_class) %>%
  autoplot(type = "heatmap") +
  labs(title="Train Confusion Matrix")
```




```
scored_test_xgb %>%
  conf_mat(response, .pred_class) %>%
  autoplot(type = "heatmap") +
  labs(title="Test Confusion Matrix")
```



Prediction

```
new_customers$cmp1 <- as.factor(new_customers$cmp1)
new_customers$cmp2 <- as.factor(new_customers$cmp2)
new_customers$cmp3 <- as.factor(new_customers$cmp3)
new_customers$cmp4 <- as.factor(new_customers$cmp4)
new_customers$cmp5 <- as.factor(new_customers$cmp5)



prediction <- predict(xgb_wflow, new_customers, type = "prob") %>%
  bind_cols(predict(xgb_wflow, new_customers, type = "class")) %>%
  bind_cols(new_customers) %>%
  dplyr:::select.data.frame(id, .pred_1, response = .pred_class)

head(prediction)
```

id	.pred_1	response
<dbl>	<dbl>	<fct>
111	0.05317521	0
112	0.10182893	0
113	0.01200235	0

id	.pred_1	response
<dbl>	<dbl>	<fct>
114	0.08175731	0
115	0.04395473	0
116	0.02377176	0

6 rows

```
prediction %>% write_csv("project_3_Xuhui Ying.csv")
```