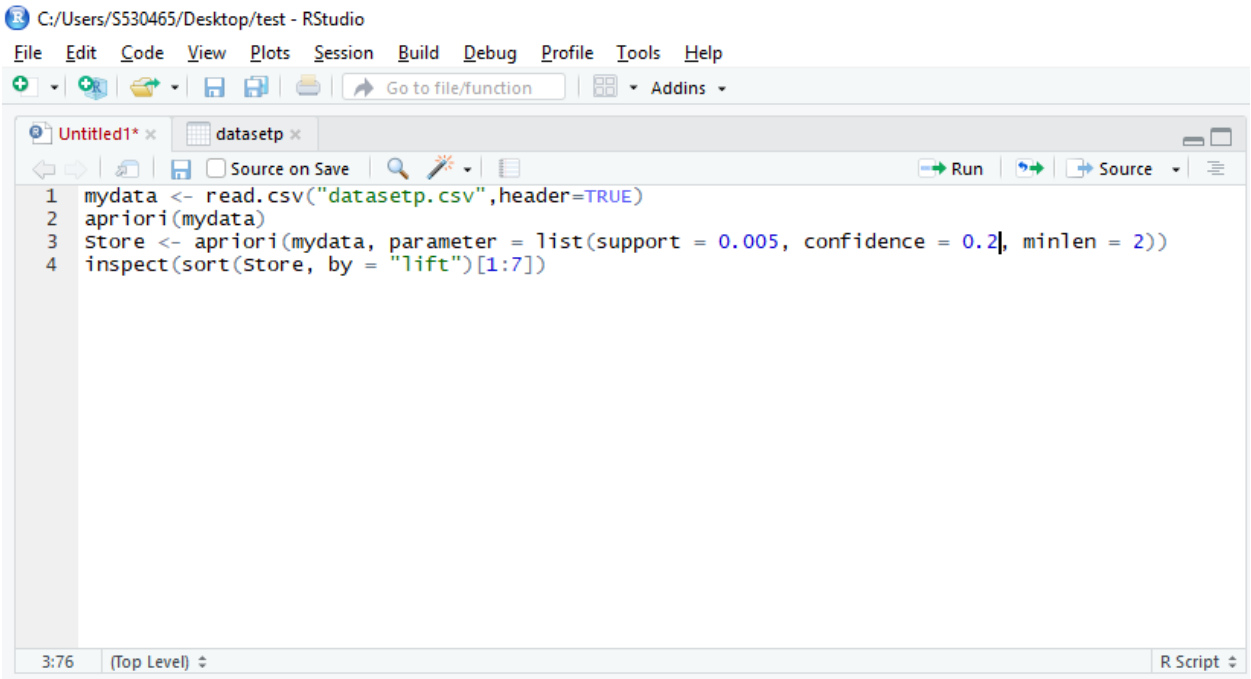
Assignment 4

1.

63	salty snack	frankfurter	apple juice	bottled beer	apple juice
64	yogurt	sausage	onions	curd	onions
65	domestic eggs	domestic eggs	onions	newspapers	onions
66	soda	organic milk	domestic eggs	rolls/buns	domestic eggs
67	soda	rolls/buns	bottled beer	newspapers	bottled beer
68	soda	organic milk	hard cheese	cookies marshmallow	hard cheese
69	rolls/buns	sausage	root vegetables	tomatoes	root vegetables
70	beverages	curd	cheese milk	blueberries	cheese milk
71	rolls/buns	frankfurter	hamburger meat	organic milk	hamburger meat
72	rolls/buns	meat pieces	cheese	newspapers	cheese
73	pip fruit	sausage	meat	tropical fruit	meat
74	beverages	pork	organic milk	whipped/sour cream	organic milk
75	pip fruit	blueberries	pastry	coffee	pastry
76	beverages	organic milk	sliced cheese	margarine	sliced cheese
77	yogurt	frankfurter	ham	organic milk	ham
78	specialty chocolate	beef	organic milk	brown bread	organic milk
79	cream cheese	pineapple	tomatoes	cheese milk	tomatoes
80	coffee	sausage	organic milk	rolls/buns	organic milk
81	pip fruit	pastry	coffee		coffee
82	beverages	meat pieces	beef	specialty cookies	beef
83	sliced cheese	root vegetables	tomatoes	organic milk	tomatoes
84	organic milk	organic milk	frozen meals	rolls/buns	frozen meals



```
Mining stopped (time limit reached). Only patterns up to a length of 5 returned!
> inspect(sort(Store, by = "lift")[1:7])
    lhs                                rhs                                support confidence   lift count
[1] {citrus.fruit=finished products} => {apple.juice=finished products}  0.005084401          1 196.68    50
[2] {apple.juice=finished products}  => {citrus.fruit=finished products} 0.005084401          1 196.68    50
[3] {citrus.fruit=finished products,
     X.12=}                        => {apple.juice=finished products}  0.005084401          1 196.68    50
[4] {apple.juice=finished products,
     X.12=}                        => {citrus.fruit=finished products} 0.005084401          1 196.68    50
[5] {citrus.fruit=finished products,
     X.13=}                        => {apple.juice=finished products}  0.005084401          1 196.68    50
[6] {apple.juice=finished products,
     X.13=}                        => {citrus.fruit=finished products} 0.005084401          1 196.68    50
[7] {citrus.fruit=finished products,
     X.14=}                        => {apple.juice=finished products}  0.005084401          1 196.68    50
> inspect(sort(Store, by = "lift")[1:5])
```

- **Apple juice and citrus fruit have been bought frequently**

2.

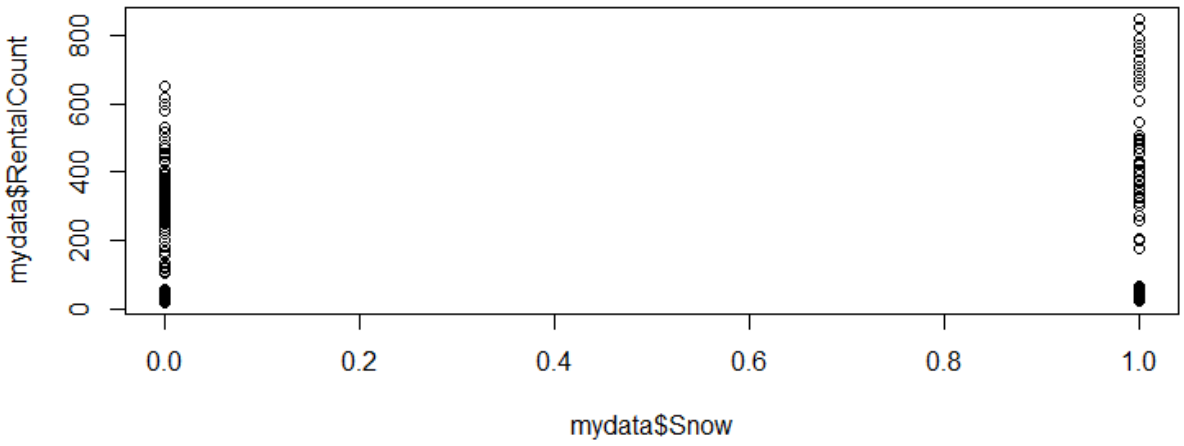
codeconsole.R*

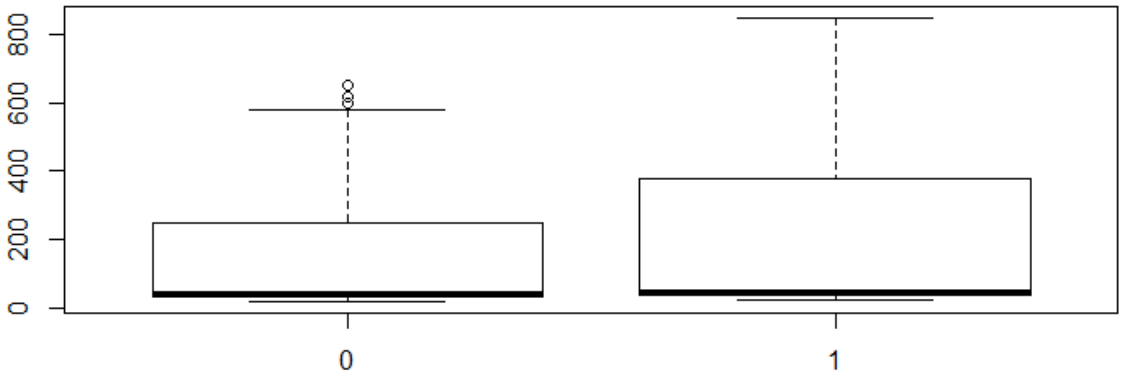
Source on Save | Run | Source

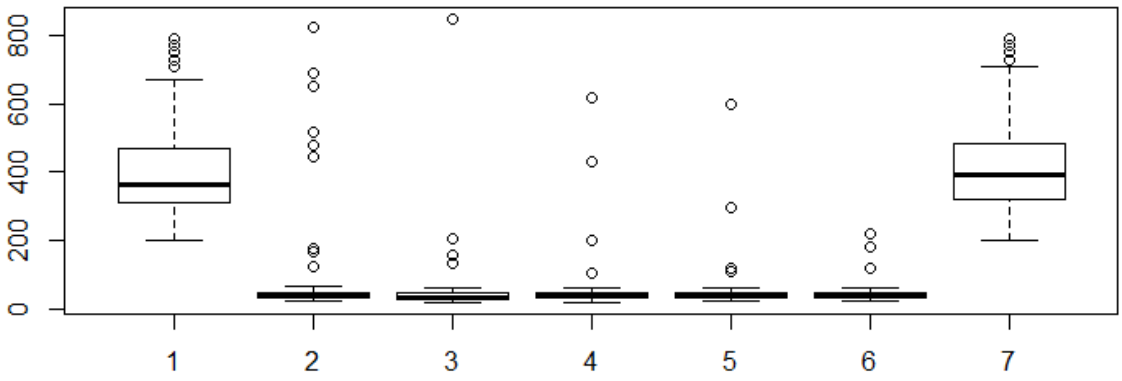
```
1  mydata =
2    read.table(".data/RentalFeatures.txt",header=TRUE)
3  colnames(mydata)
4  colnames(mydata)[1] <"Year"
5  mydata
6  head(mydata)
7  mydata$FHoliday = factor(mydata$Holiday)
8  mydata$FSnow = factor(mydata$Snow)
9  mydata$FWeekDay = factor(mydata$WeekDay)
10 train_data = mydata[mydata$Year < 2015,]
11 test_data = mydata[mydata$Year == 2015,]
12 test_counts <test_data$RentalCount
13 plot(mydata$Snow, mydata$RentalCount)
14 plot(factor(mydata$Snow), mydata$RentalCount)
15 plot(factor(mydata$WeekDay), mydata$RentalCount)
16 plot(ISOdate(mydata$Year, mydata$Month, mydata$Day),mydata$RentalCount)
17 model= lm(RentalCount ~ Month + Day + FWeekDay + FSnow +
18            FHoliday, train_data,)
19 p = predict(model, test_data)
20 plot(p-test_counts)
21 library(rpart)
22 model= rpart(RentalCount ~ Month + Day + FWeekDay +
23               FSnow + FHoliday, train_data,)
24 p = predict(model, test_data)
25 plot(p-test_counts)
26 predict(model, data.frame(Month = 1, Day = 1, FWeekDay
27                           = factor( 7), FSnow = factor(1), FHoliday = factor(0)))
28 predict(model, data.frame(Month = 1, Day = 1, FWeekDay
29                           = factor( 4), FSnow = factor(1), FHoliday = factor(0)))
```

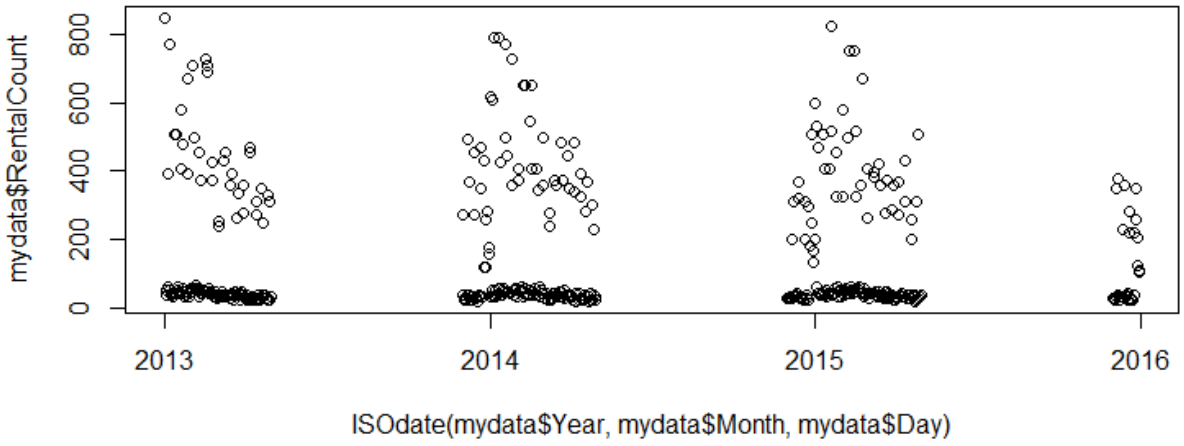
```
Console C:/Users/S530465/Desktop/Assignment04/
4 2014      3  31          38        2       0     0        0      0        2
5 2014      4  24          23        5       0     0        0      0        5
6 2015      2  11          42        4       0     0        0      0        4
> mydata$FHoliday = factor(mydata$Holiday)
> mydata$FSnow = factor(mydata$Snow)
> mydata$FWeekDay = factor(mydata$WeekDay)
> train_data = mydata[mydata$Year < 2015,]
> test_data = mydata[mydata$Year == 2015,]
> test_counts <test_data$RentalCount
  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [17] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [33] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [49] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [65] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [81] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [97] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[113] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[129] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[145] FALSE FALSE FALSE FALSE FALSE FALSE FALSE
> plot(mydata$Snow, mydata$RentalCount)
> model= lm(RentalCount ~ Month + Day + FWeekDay + FSnow +
+               FHoliday, train_data,)
> p = predict(model, test_data)
> plot(p-test_counts)
> library(rpart)
> model= rpart(RentalCount ~ Month + Day + FWeekDay +
+                  FSnow + FHoliday, train_data,)
> p = predict(model, test_data)
> plot(p-test_counts)
> predict(model, data.frame(Month = 1, Day = 1, FWeekDay
+                           = factor( 7), FSnow = factor(1), FHoliday = factor(0)))
       1
645.7059
> predict(model, data.frame(Month = 1, Day = 1, FWeekDay = factor( 4), FSnow = factor(1), FHoliday = fac
tor(0)))
       1
42.02871
>
```

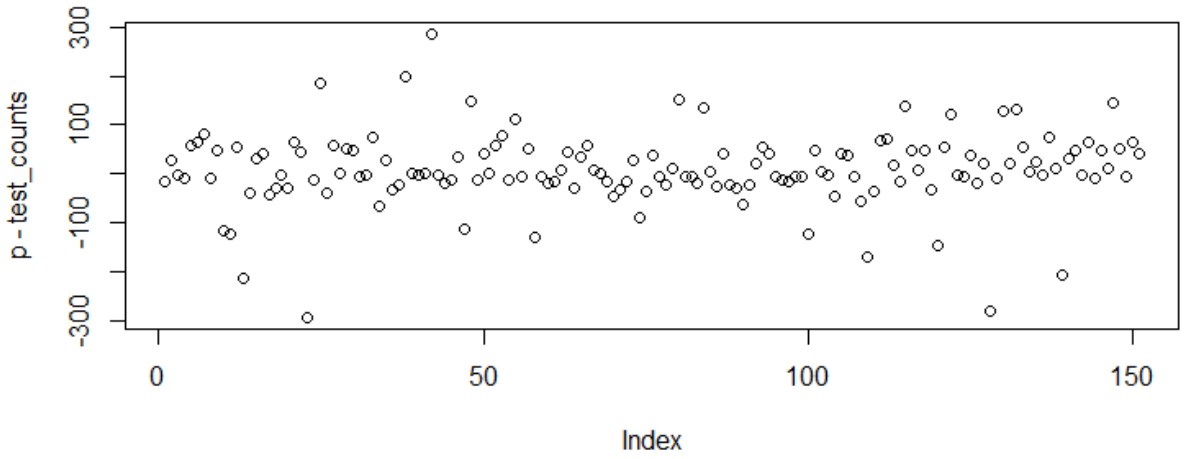
```
> predict(model, data.frame(Month = 1, Day = 1, FWeekDay
+                           = factor( 7), FSnow = factor(1), FHoliday = factor(0)))
       1
645.7059
> predict(model, data.frame(Month = 1, Day = 1, FWeekDay = factor( 4), FSnow = factor(1), FHoliday = fac
tor(0)))
       1
42.02871
```

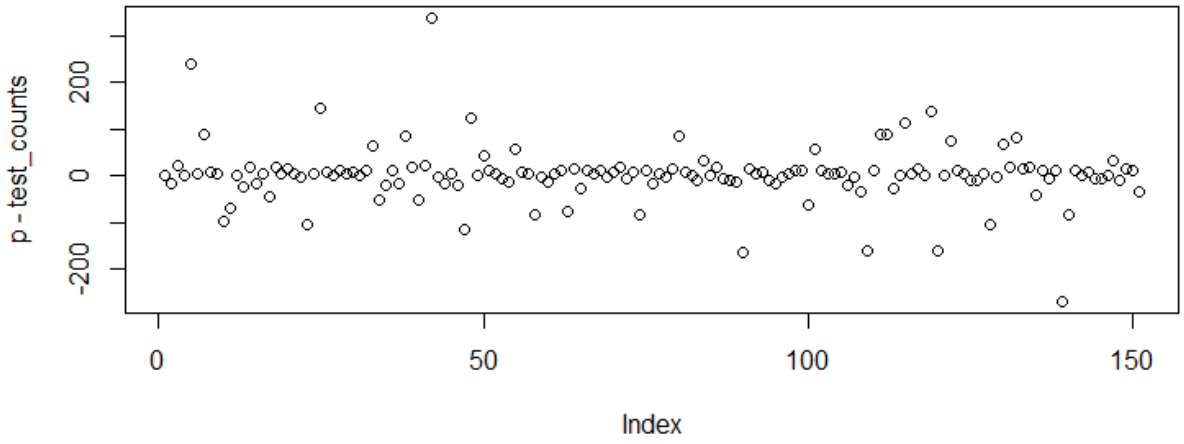



0
200
400
600
800
0
1


0
200
400
600
800
1
2
3
4
5
6
7


mydata$RentalCount
0
200
400
600
800
2013
2014
2015
2016
ISOdate(mydata$Year, mydata$Month, mydata$Day)


p - test_counts
-300
-100
100
300
0
50
100
150
Index



- on a Thursday Day(4) the output is smaller.
- on a sunday Day(7) the output is 645.7059