Linear Models

CS 4375 - Intro to Machine Learning

Dr. Karen Mazidi

Author: Leo Nguyen -ldn190002

Date: Sep 25, 2022

Linear Model for Classification (Logistic Regression)

```
- The linear model for classification let us know what class the the target variable belong too.
In most cases, it is a binary output which mean the target either belong to one or other class.
However, the model is also allow classification on more than 2 classes

- Strengths of logistic regression: its computation is inexpensive, it can separates classes wel
l if they are linearly separable, it has a nice probabilistic output

- Weaknesses of logistic regression: it is not flexible enough to capture complex non-linear dec
ision boundaries, it prone to under-fitting
```

Room Occupancy Estimation Data Set

Citation:

```
Adarsh Pal Singh, Vivek Jain, Sachin Chaudhari, Frank Alexander Kraemer, Stefan Werner and Visha
l Garg, â€œMachine Learning-Based Occupancy Estimation Using Multivariate Sensor Nodes,â€  in 20
18 IEEE Globecom Workshops (GC Wkshps), 2018.
```

Attribute Information:

```
- Date: YYYY/MM/DD
- Time: HH:MM:SS
- Temperature: In degree Celsius
- Light: In Lux
- Sound: In Volts (amplifier output read by ADC)
- CO2: In PPM
- CO2 Slope: Slope of CO2 values taken in a sliding window
- PIR: Binary value conveying motion detection
- Room_Occupancy_Count: Ground Truth
```

Load the data

```
df <- read.csv("data/Occupancy_Estimation.csv", header=TRUE)
str(df)
```

```
## 'data.frame':    10129 obs. of  19 variables:
##  $ Date                : chr  "2017/12/22" "2017/12/22" "2017/12/22" "2017/12/22" ...
##  $ Time                : chr  "10:49:41" "10:50:12" "10:50:42" "10:51:13" ...
##  $ S1_Temp             : num  24.9 24.9 25 25 25 ...
##  $ S2_Temp             : num  24.8 24.8 24.8 24.8 24.8 ...
##  $ S3_Temp             : num  24.6 24.6 24.5 24.6 24.6 ...
##  $ S4_Temp             : num  25.4 25.4 25.4 25.4 25.4 ...
##  $ S1_Light            : int  121 121 121 121 121 121 120 121 122 101 ...
##  $ S2_Light            : int  34 33 34 34 34 34 34 34 35 34 ...
##  $ S3_Light            : int  53 53 53 53 54 54 54 54 56 57 ...
##  $ S4_Light            : int  40 40 40 40 40 40 40 41 43 43 ...
##  $ S1_Sound            : num  0.08 0.93 0.43 0.41 0.18 0.13 1.39 0.09 0.09 3.84 ...
##  $ S2_Sound            : num  0.19 0.05 0.11 0.1 0.06 0.06 0.32 0.06 0.05 0.64 ...
##  $ S3_Sound            : num  0.06 0.06 0.08 0.1 0.06 0.06 0.43 0.09 0.06 0.48 ...
##  $ S4_Sound            : num  0.06 0.06 0.06 0.09 0.06 0.07 0.06 0.05 0.13 0.39 ...
##  $ S5_CO2              : int  390 390 390 390 390 390 390 390 390 390 ...
##  $ S5_CO2_Slope        : num  0.769 0.646 0.519 0.388 0.254 ...
##  $ S6_PIR              : int  0 0 0 0 0 0 1 0 0 1 ...
##  $ S7_PIR              : int  0 0 0 0 0 0 0 0 0 1 ...
##  $ Room_Occupancy_Count: int  1 1 1 1 1 1 1 1 1 1 ...
```

Data Cleaning

```
- We will remove date and time in data set. Then make PIR (in both sensor S6 and S7) and Room_Oc
cupancy_Count as factors.
```

```
df <- df[,c(3:19)]
df$S6_PIR <- factor(df$S6_PIR)
df$S7_PIR <- factor(df$S7_PIR)
df$Room_Occupancy_Count <- factor(df$Room_Occupancy_Count)
dim(df)
```

```
## [1] 10129    17
```

Divide into 80/20 train/test

```
set.seed(1234)
i <- sample(1:nrow(df), 0.8*nrow(df), replace=FALSE)
train <- df[i,]
test <- df[-i,]
```

Data Exploration

1. Look at the first 10 rows in training data

```
head(train, n=10)
```

```
##      S1_Temp S2_Temp S3_Temp S4_Temp S1_Light S2_Light S3_Light S4_Light
## 7452   25.31   25.31   24.81   25.69        0        0        0        0
## 8016   25.06   25.06   24.63   25.25        6        6       32       22
## 7162   25.38   25.38   24.94   25.81        0        0        0        0
## 8086   25.50   25.63   25.31   25.63       10       12       56       35
## 7269   25.31   25.38   24.88   25.81        0        0        0        0
## 9196   25.19   25.25   24.81   25.25        0        0        0        0
## 623    26.13   25.75   25.69   26.25      117       27      195       23
## 934    26.38   28.13   26.13   26.50      142      226      170       10
## 2948   25.69   25.38   25.19   26.00      122       31       74       53
## 2146   25.13   25.13   24.50   25.31        0        0        0        0
##      S1_Sound S2_Sound S3_Sound S4_Sound S5_CO2 S5_CO2_Slope S6_PIR S7_PIR
## 7452     0.08     0.06     0.07     0.09    355   0.00000000      0      0
## 8016     0.08     0.05     0.07     0.11    350   0.00000000      0      0
## 7162     0.07     0.05     0.07     0.10    355   0.00000000      0      0
## 8086     0.10     0.36     0.40     0.09    370   0.35384615      0      1
## 7269     0.07     0.05     0.06     0.10    355   0.00000000      0      0
## 9196     0.07     0.05     0.06     0.08    345   0.00000000      0      0
## 623      0.41     0.27     0.82     0.21    630   0.22307692      1      0
## 934      2.17     1.75     3.65     1.52   1220   0.27307692      1      1
## 2948     0.74     1.12     0.08     0.31    380   0.04615385      1      1
## 2146     0.07     0.05     0.06     0.05    360  -0.02692308      0      0
##      Room_Occupancy_Count
## 7452                    0
## 8016                    0
## 7162                    0
## 8086                    3
## 7269                    0
## 9196                    0
## 623                     2
## 934                     3
## 2948                    1
## 2146                    0
```

2. View the summary of entire training data set

```
summary(train)
```

```
##     S1_Temp         S2_Temp         S3_Temp         S4_Temp     
##  Min.   :24.94   Min.   :24.75   Min.   :24.44   Min.   :24.94  
##  1st Qu.:25.19   1st Qu.:25.19   1st Qu.:24.69   1st Qu.:25.44  
##  Median :25.38   Median :25.38   Median :24.94   Median :25.75  
##  Mean   :25.45   Mean   :25.55   Mean   :25.06   Mean   :25.75  
##  3rd Qu.:25.63   3rd Qu.:25.63   3rd Qu.:25.38   3rd Qu.:26.00  
##  Max.   :26.38   Max.   :29.00   Max.   :26.19   Max.   :26.50  
##     S1_Light          S2_Light          S3_Light          S4_Light    
##  Min.   :  0.00   Min.   :  0.00   Min.   :  0.00   Min.   : 0.00  
##  1st Qu.:  0.00   1st Qu.:  0.00   1st Qu.:  0.00   1st Qu.: 0.00  
##  Median :  0.00   Median :  0.00   Median :  0.00   Median : 0.00  
##  Mean   : 25.37   Mean   : 25.97   Mean   : 34.06   Mean   :13.34  
##  3rd Qu.: 12.00   3rd Qu.: 14.00   3rd Qu.: 50.00   3rd Qu.:22.00  
##  Max.   :165.00   Max.   :258.00   Max.   :280.00   Max.   :74.00  
##     S1_Sound          S2_Sound          S3_Sound          S4_Sound     
##  Min.   :0.0600   Min.   :0.0400   Min.   :0.0400   Min.   :0.0500  
##  1st Qu.:0.0700   1st Qu.:0.0500   1st Qu.:0.0600   1st Qu.:0.0600  
##  Median :0.0800   Median :0.0500   Median :0.0600   Median :0.0800  
##  Mean   :0.1681   Mean   :0.1205   Mean   :0.1583   Mean   :0.1033  
##  3rd Qu.:0.0800   3rd Qu.:0.0600   3rd Qu.:0.0700   3rd Qu.:0.1000  
##  Max.   :3.8800   Max.   :3.4400   Max.   :3.6700   Max.   :3.4000  
##      S5_CO2        S5_CO2_Slope     S6_PIR   S7_PIR   Room_Occupancy_Count
##  Min.   : 345.0   Min.   :-6.2962   0:7371   0:7471   0:6589              
##  1st Qu.: 355.0   1st Qu.:-0.0500   1: 732   1: 632   1: 376              
##  Median : 360.0   Median : 0.0000                     2: 589              
##  Mean   : 460.2   Mean   :-0.0119                     3: 549              
##  3rd Qu.: 465.0   3rd Qu.: 0.0000                                         
##  Max.   :1270.0   Max.   : 8.9808                                         
```

3. Count number of NA value by column

```
colSums(is.na(train))
```

```
##              S1_Temp              S2_Temp              S3_Temp 
##                    0                    0                    0 
##              S4_Temp             S1_Light             S2_Light 
##                    0                    0                    0 
##             S3_Light             S4_Light             S1_Sound 
##                    0                    0                    0 
##             S2_Sound             S3_Sound             S4_Sound 
##                    0                    0                    0 
##               S5_CO2         S5_CO2_Slope               S6_PIR 
##                    0                    0                    0 
##               S7_PIR Room_Occupancy_Count 
##                    0                    0 
```

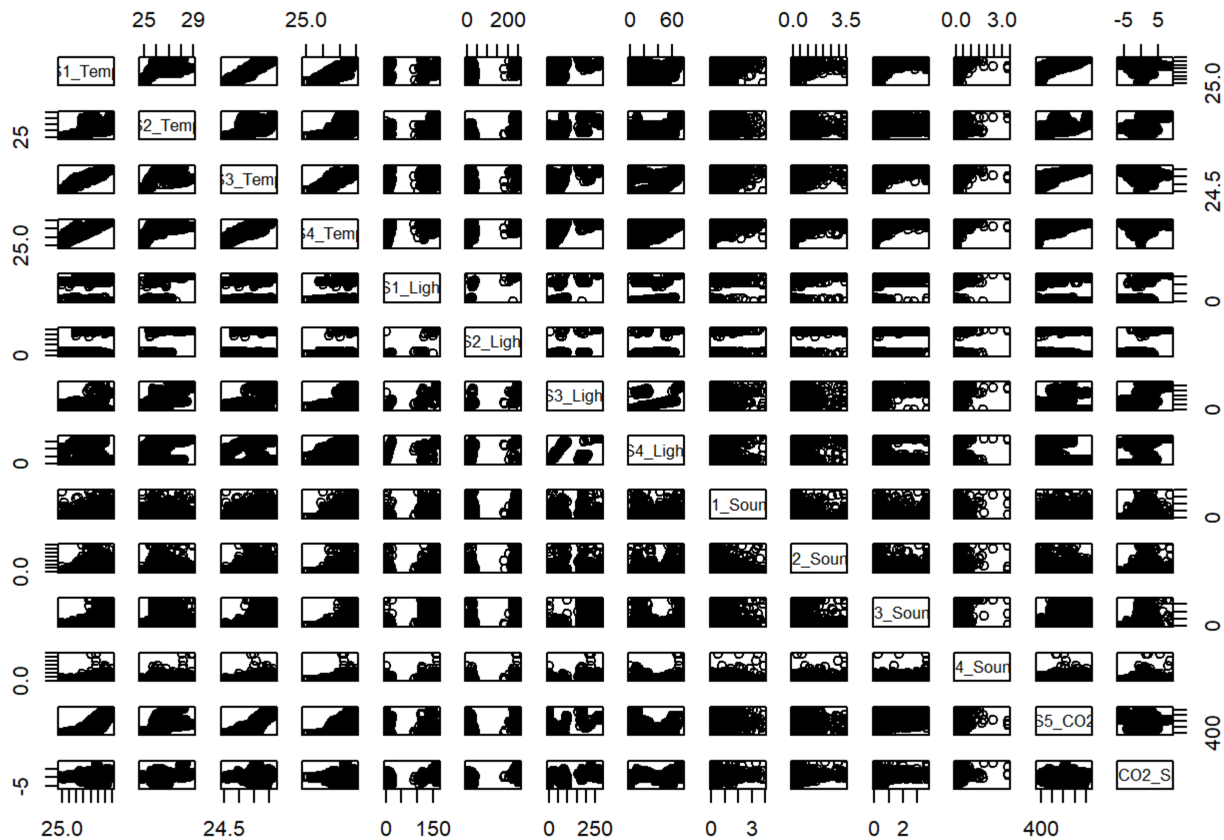
4. Check the correlation of all non-factored column in training data

```
cor(train[1:14])
```

```
##                S1_Temp   S2_Temp    S3_Temp   S4_Temp  S1_Light  S2_Light
## S1_Temp      1.0000000 0.7988126 0.94854178 0.8561668 0.6772522 0.5491254
## S2_Temp      0.7988126 1.0000000 0.76338956 0.6964297 0.6423796 0.6492157
## S3_Temp      0.9485418 0.7633896 1.00000000 0.8857518 0.5896138 0.4992400
## S4_Temp      0.8561668 0.6964297 0.88575176 1.0000000 0.5810648 0.4584023
## S1_Light     0.6772522 0.6423796 0.58961377 0.5810648 1.0000000 0.8424217
## S2_Light     0.5491254 0.6492157 0.49924000 0.4584023 0.8424217 1.0000000
## S3_Light     0.6398829 0.6092496 0.63880023 0.5860075 0.8137460 0.7133490
## S4_Light     0.2171919 0.3776189 0.30554561 0.3906857 0.5170222 0.4647888
## S1_Sound     0.4326500 0.4408342 0.37233437 0.3520467 0.5988507 0.5089754
## S2_Sound     0.3827064 0.4036968 0.33528651 0.3065751 0.5238735 0.5561147
## S3_Sound     0.4368825 0.4271445 0.39600751 0.3398152 0.4929767 0.4479513
## S4_Sound     0.3403014 0.3711698 0.31142271 0.2831699 0.4292521 0.4167285
## S5_CO2       0.8641144 0.7400169 0.81905401 0.6486232 0.5960824 0.5664115
## S5_CO2_Slope 0.1334090 0.2004125 0.09090632 0.1039258 0.5021404 0.4950837
##               S3_Light  S4_Light  S1_Sound  S2_Sound  S3_Sound  S4_Sound
## S1_Temp      0.6398829 0.2171919 0.4326500 0.3827064 0.4368825 0.3403014
## S2_Temp      0.6092496 0.3776189 0.4408342 0.4036968 0.4271445 0.3711698
## S3_Temp      0.6388002 0.3055456 0.3723344 0.3352865 0.3960075 0.3114227
## S4_Temp      0.5860075 0.3906857 0.3520467 0.3065751 0.3398152 0.2831699
## S1_Light     0.8137460 0.5170222 0.5988507 0.5238735 0.4929767 0.4292521
## S2_Light     0.7133490 0.4647888 0.5089754 0.5561147 0.4479513 0.4167285
## S3_Light     1.0000000 0.5860602 0.5025905 0.4236222 0.5770816 0.4556217
## S4_Light     0.5860602 1.0000000 0.3001558 0.3046095 0.1720692 0.1990725
## S1_Sound     0.5025905 0.3001558 1.0000000 0.5680242 0.5349285 0.5560886
## S2_Sound     0.4236222 0.3046095 0.5680242 1.0000000 0.5187057 0.5746383
## S3_Sound     0.5770816 0.1720692 0.5349285 0.5187057 1.0000000 0.6791243
## S4_Sound     0.4556217 0.1990725 0.5560886 0.5746383 0.6791243 1.0000000
## S5_CO2       0.6480523 0.1508468 0.3913346 0.3221456 0.4458916 0.3179838
## S5_CO2_Slope 0.4464251 0.2089222 0.3414812 0.3537798 0.3231514 0.3248221
##                  S5_CO2 S5_CO2_Slope
## S1_Temp      0.86411441   0.13340898
## S2_Temp      0.74001695   0.20041246
## S3_Temp      0.81905401   0.09090632
## S4_Temp      0.64862318   0.10392584
## S1_Light     0.59608242   0.50214040
## S2_Light     0.56641149   0.49508366
## S3_Light     0.64805232   0.44642514
## S4_Light     0.15084684   0.20892220
## S1_Sound     0.39133463   0.34148117
## S2_Sound     0.32214555   0.35377982
## S3_Sound     0.44589156   0.32315140
## S4_Sound     0.31798384   0.32482214
## S5_CO2       1.00000000   0.06830595
## S5_CO2_Slope 0.06830595   1.00000000
```

5. Quick visualization on correlation

```
pairs(train[1:14])
```



6. Check the level, encoding matrix of “Room_Occupancy_Count” in training data

Check Level

```
levels(train$Room_Occupancy_Count)
```

```
## [1] "0" "1" "2" "3"
```

Check Encoding Matrix

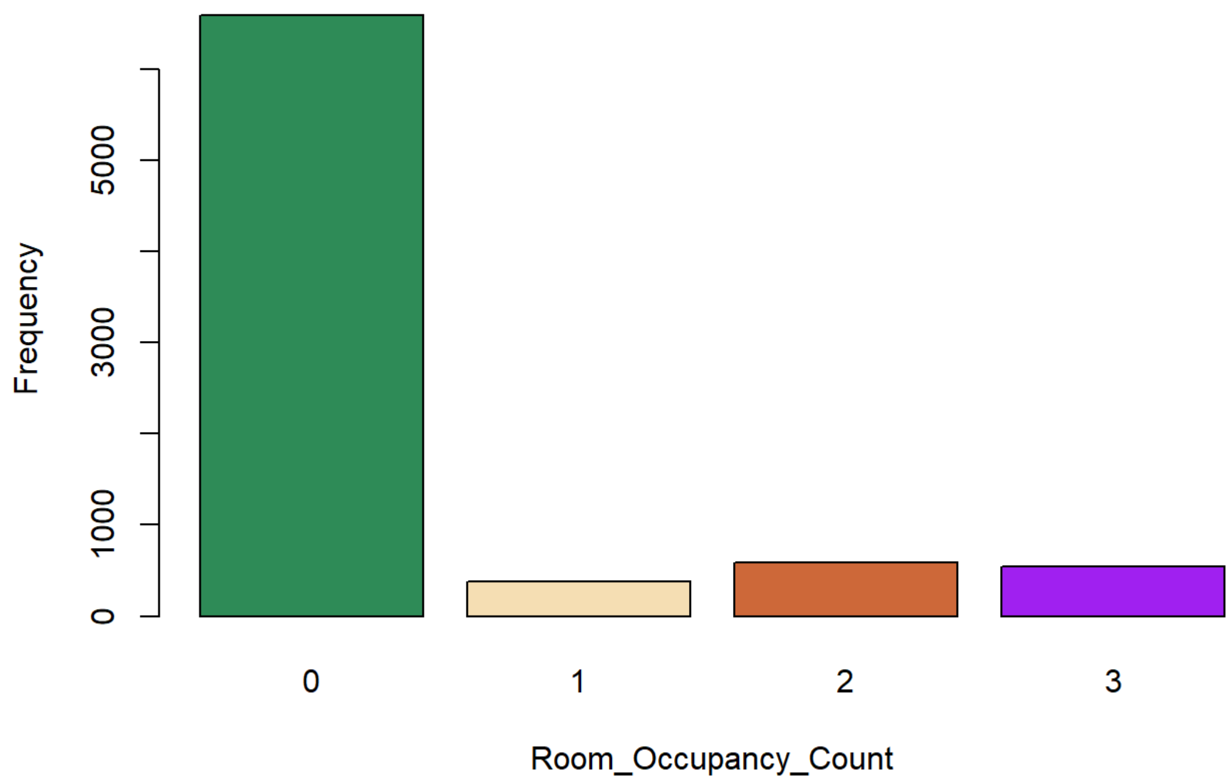
```
contrasts(train$Room_Occupancy_Count)
```

```
##   1 2 3
## 0 0 0 0
## 1 1 0 0
## 2 0 1 0
## 3 0 0 1
```

Data Visualization using informative graph

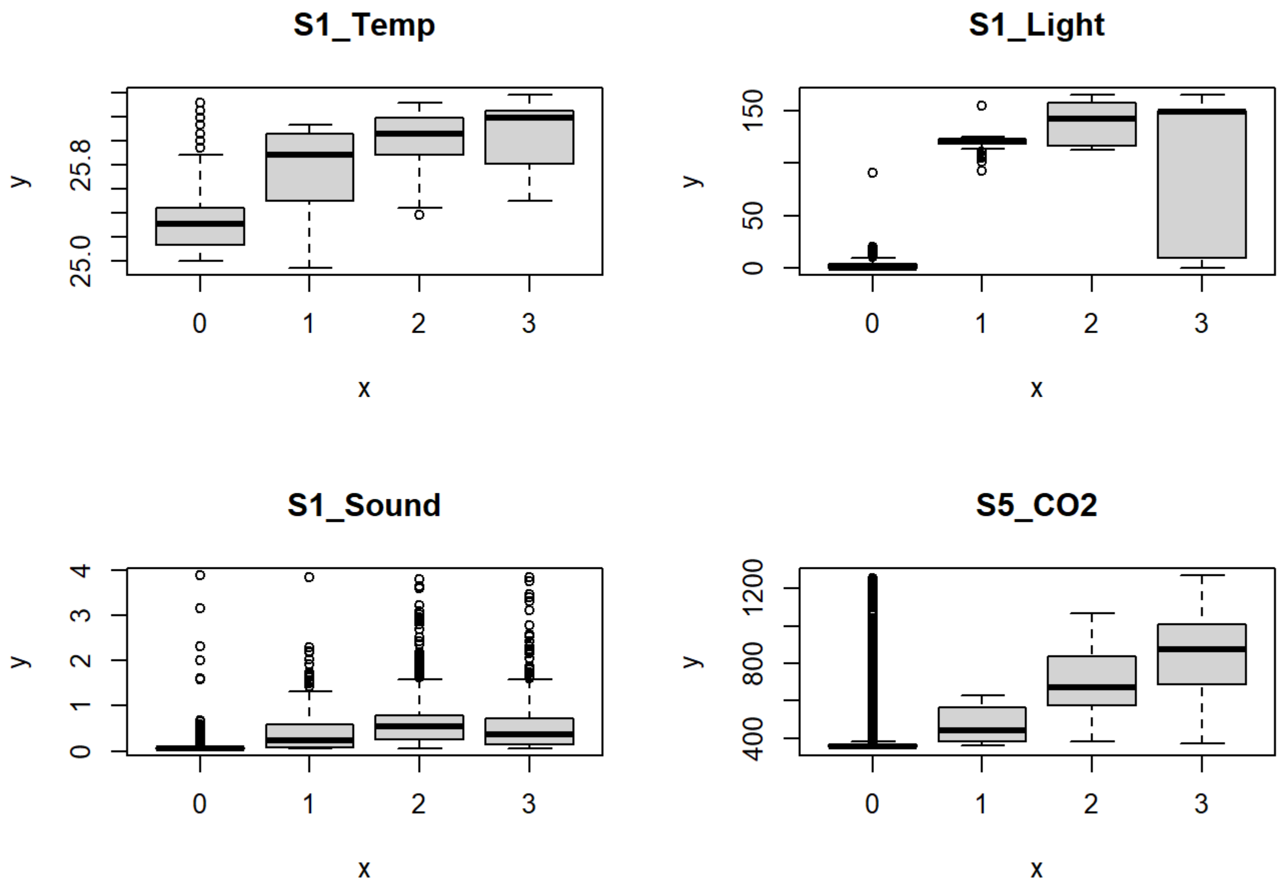
1. Using barplots to visualize the 4 factor level of “Room_Occupancy_Count”

```
counts <- table(train$Room_Occupancy_Count)
barplot(counts, xlab="Room_Occupancy_Count", ylab="Frequency", col=c("seagreen","wheat","sienna
3", "purple"))
```



2. Visualize the relationship between “Room_Occupancy_Count” and Temp, Light, Sound, CO2
    - We can see clearly that the number of people in the room increase when the Temp and CO2 level increase. Not so much with Light and Sound

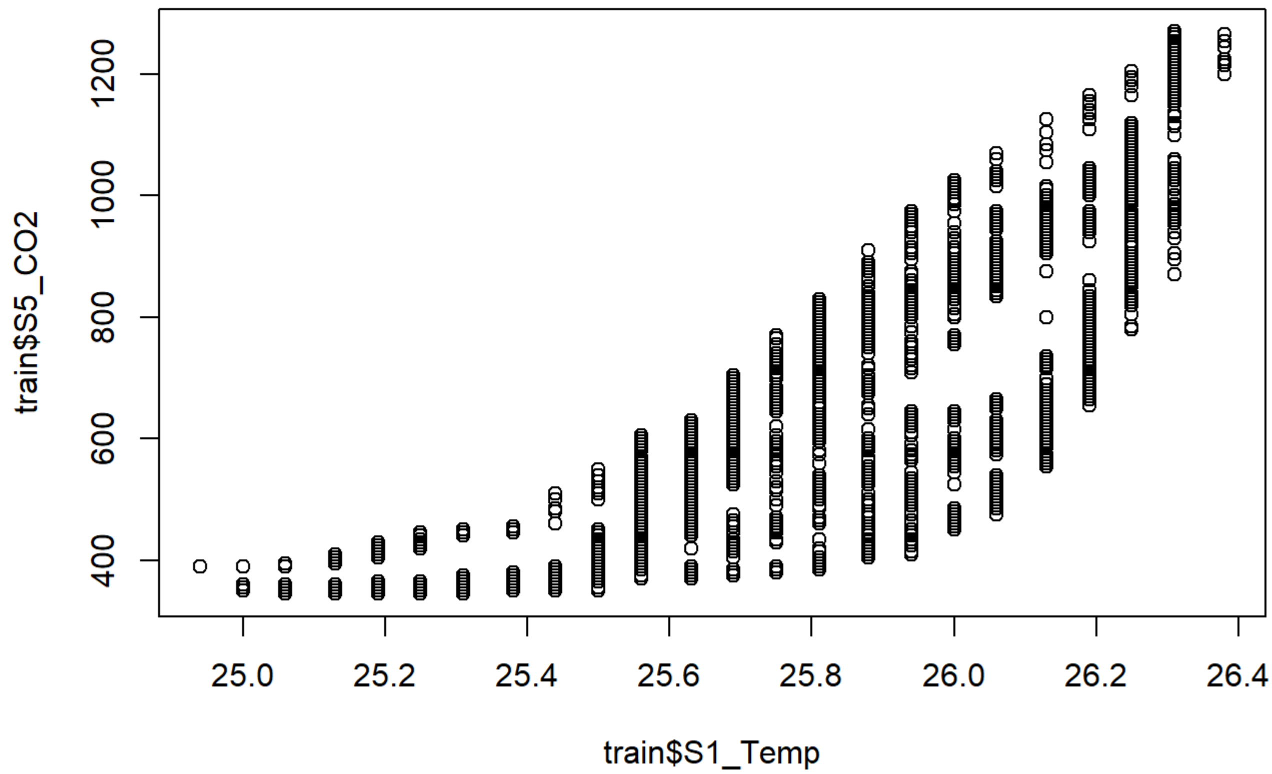
```
par(mfrow=c(2,2))
plot(train$Room_Occupancy_Count, train$S1_Temp, data = train, main = "S1_Temp")
plot(train$Room_Occupancy_Count, train$S1_Light, data = train, main = "S1_Light")
plot(train$Room_Occupancy_Count, train$S1_Sound, data = train, main = "S1_Sound")
plot(train$Room_Occupancy_Count, train$S5_CO2, data = train, main = "S5_CO2")
```



3. The relationship between the CO2 Level and Room Temp

- More people inside the room, the more CO2, and higher Temp

```
plot(train$S5_CO2~train$S1_Temp)
```



Build Logistic Regression Model

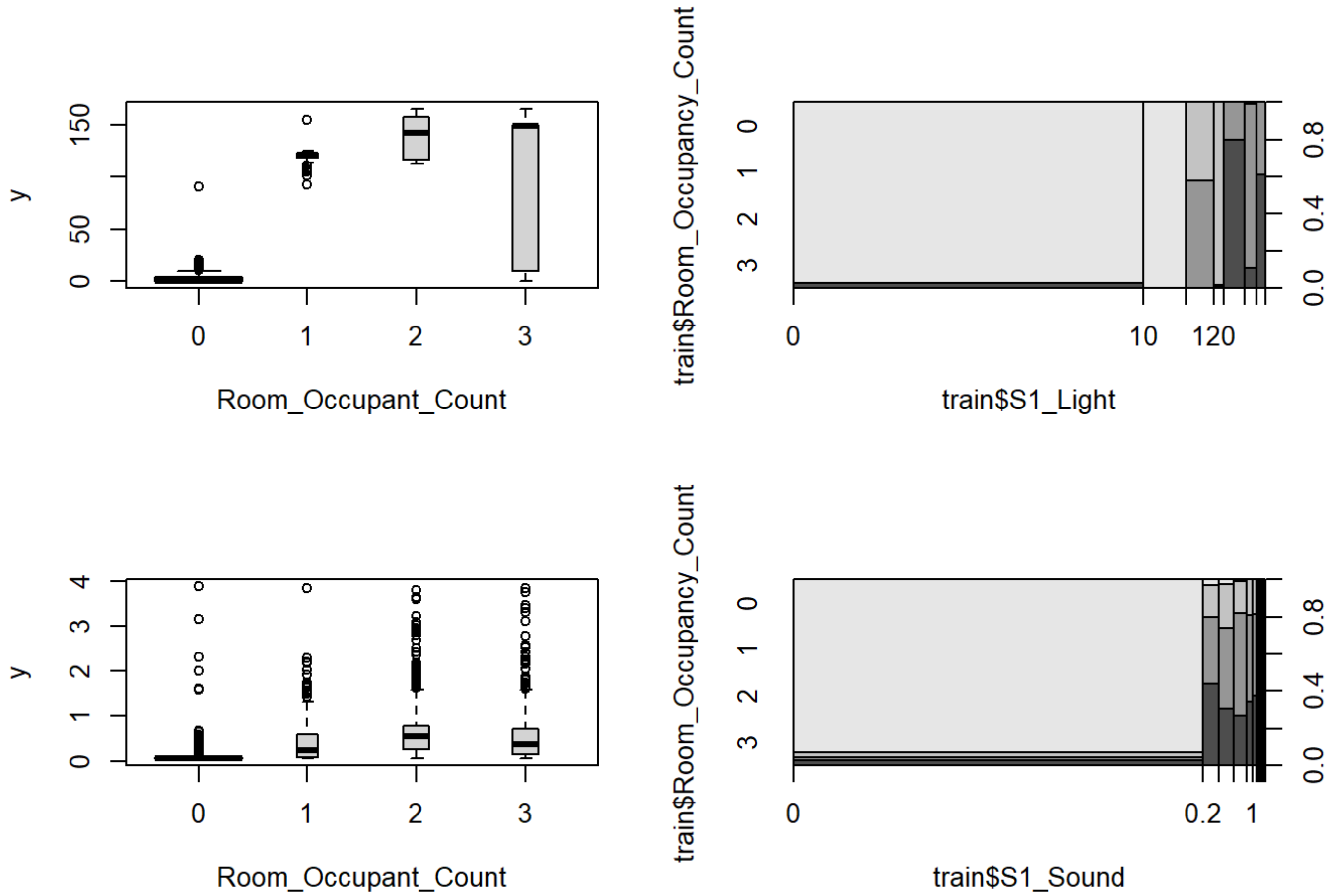
```
- We remove Sound(S1-S4) and Light(S1,S2) because the model will issue the "Warning: glm.fit: al
gorithm did not converge. Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred" wh
ich mean the data was separated perfectly --> This is not very good, in fact, it make all the pr
edictors become unreliable (not 3 asterisks on any of predictors if we keep those). Those Sound
(S1-S4) and Light(S1,S2) create a perfect separation between zero occupants and at least 1 occup
ant.

- In case of Sound(S1-S4): it makes sense because, it is obvious that if there no one in the roo
m, properly, there is no sound or low sound level. But when there is people in the room, there w
ill be sound. The problem is we can not distinguish on how many people are there.
- In case of Light(S1, S2): it could be because the location of the sensor is not optimal. Their
positions are always cover when someone in the room. So it can only distingush between room empt
y and not empty. And can not account for how many people are there.
```

We can visual the problem with plot below

```
par(mfrow=c(2,2))
plot(train$Room_Occupancy_Count, train$S1_Light, xlab="Room_Occupant_Count", varwidth=TRUE, data
= train)
plot(train$Room_Occupancy_Count~train$S1_Light, data = train)

plot(train$Room_Occupancy_Count, train$S1_Sound, xlab="Room_Occupant_Count", varwidth=TRUE, data
= train)
plot(train$Room_Occupancy_Count~train$S1_Sound, data = train)
```




```
- So we remove those column in the model. It results in a better model, all the predictors now h
ave 3 asterisks.
```

```
glm1 <- glm(Room_Occupancy_Count~.-S1_Sound-S2_Sound-S3_Sound-S4_Sound-S1_Light-S2_Light, data =
train, family="binomial")
summary(glm1)
```

```
##
## Call:
## glm(formula = Room_Occupancy_Count ~ . - S1_Sound - S2_Sound -
##     S3_Sound - S4_Sound - S1_Light - S2_Light, family = "binomial",
##     data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -5.2729  -0.1199  -0.0683  -0.0301   3.4389
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -2.295e+02  1.367e+01 -16.793  < 2e-16 ***
## S1_Temp       2.239e+01  1.286e+00  17.409  < 2e-16 ***
## S2_Temp      -1.721e+00  2.308e-01  -7.454 9.05e-14 ***
## S3_Temp      -5.142e+00  7.213e-01  -7.130 1.01e-12 ***
## S4_Temp      -6.525e+00  7.450e-01  -8.759  < 2e-16 ***
## S3_Light      5.975e-02  6.164e-03   9.693  < 2e-16 ***
## S4_Light      3.754e-02  9.109e-03   4.121 3.77e-05 ***
## S5_CO2       -1.211e-02  1.153e-03 -10.500  < 2e-16 ***
## S5_CO2_Slope  1.571e+00  1.120e-01  14.034  < 2e-16 ***
## S6_PIR1       4.719e+00  4.126e-01  11.437  < 2e-16 ***
## S7_PIR1       2.101e+00  5.068e-01   4.146 3.38e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 7805.05  on 8102  degrees of freedom
## Residual deviance:  940.48  on 8092  degrees of freedom
## AIC: 962.48
##
## Number of Fisher Scoring iterations: 9
```

Logistic Regression Model Summary Explanation

```
 - The model was build to predict the target column "Room_Occupation_Count with all other column
excepts the Light (S1, S2) and Sound(S1-S4) using the training data which contain 80% observatio
ns from original data set.

- All the predictors in the Coefficients section has 3 asterisks which indicate they are good pr
edictors. All the p-value is relative small too --> good indicators

- The Null deviance mean the lack of fit of the model considering only the intercept. The Residu
al deviance mean the lack of fit of the entire model. As we can see that the Residual deviance i
s much lower than the Null deviance indicates that this is a good model. We also have a very hig
h AIC number which consider as good indicator. The AIC is very useful when comparing between the
model
```

Naive Bayes

Import the necessary package e1071

```
library(e1071)
```

Build the Naive Bayes Model

```
nb1 <- naiveBayes(Room_Occupancy_Count~., data = train)
nb1
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##          0          1          2          3
## 0.81315562 0.04640257 0.07268913 0.06775268
##
## Conditional probabilities:
##    S1_Temp
## Y       [,1]      [,2]
##   0 25.33800 0.2419764
##   1 25.74003 0.3313588
##   2 26.00177 0.2289504
##   3 26.06122 0.2548582
##
##    S2_Temp
## Y       [,1]      [,2]
##   0 25.37334 0.3354533
##   1 25.84620 0.7080875
##   2 26.22947 0.6822780
##   3 26.73066 0.8050388
##
##    S3_Temp
## Y       [,1]      [,2]
##   0 24.93059 0.3323818
##   1 25.29500 0.3305083
##   2 25.58900 0.3180727
##   3 25.82485 0.2159190
##
##    S4_Temp
## Y       [,1]      [,2]
##   0 25.66040 0.3096895
##   1 26.08981 0.2802527
##   2 26.18655 0.1741173
##   3 26.18450 0.2658604
##
##    S1_Light
## Y         [,1]      [,2]
##   0   2.702383  5.185821
##   1 120.255319  4.311686
##   2 136.777589 20.338708
##   3 112.978142 65.159850
##
##    S2_Light
## Y         [,1]       [,2]
##   0   3.069054   5.932677
##   1  31.547872  15.943161
##   2 137.867572 110.050975
```

```
##   3 177.014572 103.372050
##
##    S3_Light
## Y         [,1]     [,2]
##   0  13.44438 24.34054
##   1  53.72872 15.27772
##   2 137.87946 62.87435
##   3 156.60656 79.90191
##
##    S4_Light
## Y         [,1]     [,2]
##   0  9.281985 16.86467
##   1 37.531915 11.35084
##   2 31.397284 23.59829
##   3 26.129326 22.95957
##
##    S1_Sound
## Y          [,1]       [,2]
##   0 0.07739718 0.07821733
##   1 0.42287234 0.45578269
##   2 0.65797963 0.58961145
##   3 0.55646630 0.58612414
##
##    S2_Sound
## Y          [,1]       [,2]
##   0 0.05293823 0.07044045
##   1 0.13414894 0.12301466
##   2 0.54032258 0.59060277
##   3 0.47125683 0.53143009
##
##    S3_Sound
## Y          [,1]       [,2]
##   0 0.06439369 0.08269235
##   1 0.12699468 0.10191897
##   2 0.64679117 0.82228468
##   3 0.78264117 0.99266826
##
##    S4_Sound
## Y          [,1]       [,2]
##   0 0.07974503 0.03141225
##   1 0.09351064 0.04941163
##   2 0.23499151 0.19514919
##   3 0.25204007 0.34522688
##
##    S5_CO2
## Y        [,1]      [,2]
##   0 405.1138 136.57888
##   1 468.4441  89.04927
##   2 707.7929 193.35091
##   3 850.5647 249.40522
##
##    S5_CO2_Slope
```

```
## Y         [,1]      [,2]
##   0 -0.3180919 0.8505698
##   1  0.3466653 0.9051824
##   2  1.1580188 1.2668854
##   3  2.1621900 1.3912824
##
##    S6_PIR
## Y             0           1
##   0 0.996964638 0.003035362
##   1 0.646276596 0.353723404
##   2 0.509337861 0.490662139
##   3 0.471766849 0.528233151
##
##    S7_PIR
## Y             0           1
##   0 0.998482319 0.001517681
##   1 0.957446809 0.042553191
##   2 0.570458404 0.429541596
##   3 0.357012750 0.642987250
```

Naive Bayes Model Output Explanation

- We can see the probabilities for each class 0,1,2,3 in Room_Occupant_Count are 81%, 5%, 7%, 7%. It mean the model can easily detect if the room is empty or not but not good at distinguish on how many people in the room
- For the 2 factored predictors: S6_PIR and S7_PIR, the probabilities values confirm the statement above, we have a very high value above 90% to detect of the room is empty or have at least 1 person. But the value reduce significantly when we want to verify how many people inside the room.
- For other predictors, because they are numeric, so they does not provide much information.

Predict and evaluate test data

Predict and evaluate test data using logistic regression

```
probs <- predict(glm1, newdata=test, type="response")
pred <- ifelse(probs>0.5, 1, 0)
acc <- mean(pred==test$Room_Occupancy_Count)
print(paste("accuracy = ", acc))
```

```
## [1] "accuracy =  0.834649555774926"
```

```
table(pred, test$Room_Occupancy_Count)
```

```
##
## pred    0    1    2    3
##    0 1626   18    8    1
##    1   13   65  151  144
```

Predict and evaluate test data using confusionMatrix()

```
    - The warning show up because, we count the number of each level of Room_Occupancy_Count by proparbilities with 50% threshold. And this column factor has 4 levels
```

```r
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```r
confusionMatrix(as.factor(pred), reference=test$Room_Occupancy_Count)
```

```
## Warning in confusionMatrix.default(as.factor(pred), reference =
## test$Room_Occupancy_Count): Levels are not in the same order for reference and
## data. Refactoring data to match.
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1    2    3
##          0 1626   18    8    1
##          1   13   65  151  144
##          2    0    0    0    0
##          3    0    0    0    0
##
## Overall Statistics
##
##                Accuracy : 0.8346
##                  95% CI : (0.8177, 0.8506)
##     No Information Rate : 0.809
##     P-Value [Acc > NIR] : 0.001552
##
##                   Kappa : 0.5026
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 0 Class: 1 Class: 2 Class: 3
## Sensitivity            0.9921  0.78313  0.00000  0.00000
## Specificity            0.9302  0.84148  1.00000  1.00000
## Pos Pred Value         0.9837  0.17426      NaN      NaN
## Neg Pred Value         0.9651  0.98911  0.92152  0.92843
## Prevalence             0.8090  0.04097  0.07848  0.07157
## Detection Rate         0.8026  0.03208  0.00000  0.00000
## Detection Prevalence   0.8159  0.18411  0.00000  0.00000
## Balanced Accuracy      0.9612  0.81231  0.50000  0.50000
```

Evaluate using ROC

```
- Not Applicable, as ROCR only support binary classification. The "Room_Occupancy_Count" has 4 c
lasses
```

Predict and evaluate test data using Naive Bayes

```
p1 <- predict(nb1, newdata=test, type="class")

acc <- mean(p1==test$Room_Occupancy_Count)
print(paste("accuracy = ", acc))
```

```
## [1] "accuracy =  0.95360315893386"
```

```
table(p1, test$Room_Occupancy_Count)
```

```
##     
## p1      0    1    2    3
##   0 1608    0    0   11
##   1    0   76    0    0
##   2    0    7  147   33
##   3   31    0   12  101
```

Compare result between logistic model glm1 and Naive Bayes nb1

- As we can see the Naive Bayes-nb1 have higher accuracy compare to logistic regression-glm1, and confusionMatrix (95% > 83% , 83%). This result happen because we know that Naive Bayes perform better in high dimensions (More than 2 classification classes) and our target was separated into 4 different classes.

Strengths and Weakness of Naive Bayes and Logistic Regression

Strengths and weakness of Naive Bayes

- Strengths: It work will with small data sets, it is easy to implement and interpret, it can manage high dimensions well
- Weakness: It may be outperformed by other classifiers for larger data sets

Strengths and weakness of Logistic Regression

- Strengths: its computation is inexpensive, it can separates classes well if they are linearly separable, it has a nice probabilistic output
- Weaknesses: it is not flexible enough to capture complex non-linear decision boundaries, it prone to under-fitting

Benefits and drawback of each classification metric

For Logistic Regression

```
- The benefits is that it is quick and have a nice probabilistic output. However, as we can see in the result, because, the target is 4 levels so that did not perform very well
.
```

For confusionMatrix

```
- The benefits is that it provide extra infomation on the output - Kappa value. In this example, we have Kappa = 0.5026 which is moderate agreement on quantifying between the predictors and the actual value. However, the drawback is it provide the accuracy is almost the same as Logistic Regression
```

For ROCR

```
- Not Applicable, as it only support on binary classifications
```

For Naive Bayes

```
- The benefit is that it provide the highest accuracy among classification metrics because it perform well in high dimension class. It is also easy to interpret. However, the drawback is it does not perform well in large data set
```