Multiple Regression, Cross Validation MMRE & PRED

Mohit, Songyh and AtharvaKale

3/6/2018

Loading data form the csv file

```
data <- read.csv("/Users/mohit/Development/My Scripts/modelsEvaluation.csv",stringsAsFactors= T)
summary(data)
```

```
##       NUM
##  Min.   : 1.00
##  1st Qu.: 5.75
##  Median :10.50
##  Mean   :10.50
##  3rd Qu.:15.25
##  Max.   :20.00
##
##                                                      PROJ
##  f14a_cash_doctor                                      : 1
##  f14a_gotrla                                           : 1
##  F14a_mobile_application_for_mobile_controlled_lighting: 1
##  F14a_REFERsy                                          : 1
##  F14a_sharethetraining_com                             : 1
##  F14a_soccer_data_web_crawler                          : 1
##  (Other)                                               :14
##      Effort         Effort_ALY  Effort_Norm      Norm_Factor
##  Min.   : 140.5   Min.   :0    Min.   : 206.6   Min.   :0.6700
##  1st Qu.: 294.0   1st Qu.:0    1st Qu.: 561.3   1st Qu.:0.8975
##  Median : 730.5   Median :0    Median : 831.8   Median :1.0950
##  Mean   :1064.3   Mean   :0    Mean   :1576.7   Mean   :1.1965
##  3rd Qu.:1414.6   3rd Qu.:0    3rd Qu.:2504.1   3rd Qu.:1.4275
##  Max.   :3680.0   Max.   :0    Max.   :5850.4   Max.   :1.8900
##
##      KSLOC          UEUCW_ALY       UEXUCW_ALY       UDUCW_ALY
##  Min.   : 0.552   Min.   : 75.0   Min.   : 42.00   Min.   : 29.00
##  1st Qu.: 2.777   1st Qu.:120.0   1st Qu.: 96.75   1st Qu.: 87.75
##  Median : 4.920   Median :157.5   Median :174.50   Median :148.50
##  Mean   : 5.507   Mean   :225.0   Mean   :193.10   Mean   :191.70
##  3rd Qu.: 7.422   3rd Qu.:292.5   3rd Qu.:250.50   3rd Qu.:236.50
##  Max.   :21.344   Max.   :705.0   Max.   :701.00   Max.   :777.00
##
##       UAW            TCF            TCF_ALY        EF             EF_ALY
##  Min.   : 3.00   Min.   :0.7950   Min.   :0   Min.   :0.8750   Min.   :0
##  1st Qu.: 6.00   1st Qu.:0.8738   1st Qu.:0   1st Qu.:0.9463   1st Qu.:0
##  Median : 9.00   Median :0.9200   Median :0   Median :1.0250   Median :0
##  Mean   : 8.55   Mean   :0.9280   Mean   :0   Mean   :1.0265   Mean   :0
##  3rd Qu.: 9.00   3rd Qu.:0.9363   3rd Qu.:0   3rd Qu.:1.0288   3rd Qu.:0
##  Max.   :14.00   Max.   :1.1750   Max.   :0   Max.   :1.3250   Max.   :0
##
##     EUCP_ALY         EXUCP_ALY          DUCP_ALY        Effort_Norm_UCP
##  Min.   :  77.92   Min.   :  61.57   Min.   :  49.26   Min.   : 194.1
##  1st Qu.: 117.55   1st Qu.:  94.24   1st Qu.:  94.71   1st Qu.: 528.4
##  Median : 151.75   Median : 161.15   Median : 135.31   Median : 681.4
##  Mean   : 243.80   Mean   : 200.49   Mean   : 201.15   Mean   :1165.7
##  3rd Qu.: 279.56   3rd Qu.: 215.71   3rd Qu.: 206.74   3rd Qu.:1745.5
##  Max.   :1067.00   Max.   :1061.06   Max.   :1173.84   Max.   :3265.0
##
##     Path_Num       UseCase_Num    Diagram_Num        INT
##  Min.   : 19.00   Min.   : 5.0   Min.   : 5.0   0      :4
##  1st Qu.: 33.50   1st Qu.: 8.0   1st Qu.:10.0   13     :2
##  Median : 53.00   Median :10.5   Median :12.0   30     :2
##  Mean   : 62.55   Mean   :15.0   Mean   :16.7   15     :1
##  3rd Qu.: 76.00   3rd Qu.:19.5   3rd Qu.:21.0   16     :1
##  Max.   :246.00   Max.   :47.0   Max.   :47.0   18     :1
##                                                 (Other):9
##     INT_ALY             DM        DM_ALY           CTRL
##  Min.   :  2.00   0      :4   Min.   : 8.00   0      : 4
##  1st Qu.: 13.75   10     :2   1st Qu.:12.50   51     : 2
##  Median : 25.50   13     :2   Median :17.50   18     : 1
##  Mean   : 29.50   18     :2   Mean   :23.15   26     : 1
##  3rd Qu.: 36.25   21     :2   3rd Qu.:30.75   28     : 1
##  Max.   :119.00   9      :2   Max.   :57.00   30     : 1
##                   (Other):6                   (Other):10
##     CTRL_ALY            EXTIVK     EXTIVK_ALY         EXTCLL
##  Min.   : 17.00   0         :11   Min.   :0.00   0        :13
##  1st Qu.: 28.50   1         : 2   1st Qu.:0.00   1        : 3
##  Median : 49.50   3         : 1   Median :1.00   2        : 2
##  Mean   : 52.85   5         : 4   Mean   :1.50   4        : 1
##  3rd Qu.: 71.25   6         : 1   3rd Qu.:2.25   undefined: 1
##  Max.   :168.00   undefined: 1   Max.   :6.00
##
##    EXTCLL_ALY            NT        NT_ALY          NWT_ALY
##  Min.   : 0.00   undefined:5   Min.   : 17.00   Min.   : 118.0
##  1st Qu.: 0.00   32       :2   1st Qu.: 28.50   1st Qu.: 232.5
##  Median : 0.00   51       :2   Median : 49.50   Median : 329.0
##  Mean   : 1.35   26       :1   Mean   : 52.85   Mean   : 520.5
##  3rd Qu.: 2.00   29       :1   3rd Qu.: 71.25   3rd Qu.: 570.2
##  Max.   :11.00   30       :1   Max.   :168.00   Max.   :2332.0
##                  (Other)  :8
##    NWT_DE_ALY          DET         RET             ILF         EIF
##  Min.   : 116.0   Min.   :0   Min.   :0.00   Min.   :0   Min.   :0
##  1st Qu.: 233.5   1st Qu.:0   1st Qu.:0.00   1st Qu.:0   1st Qu.:0
##  Median : 328.0   Median :0   Median :2.00   Median :0   Median :0
##  Mean   : 536.8   Mean   :0   Mean   :1.65   Mean   :0   Mean   :0
##  3rd Qu.: 581.5   3rd Qu.:0   3rd Qu.:2.00   3rd Qu.:0   3rd Qu.:0
##  Max.   :2435.0   Max.   :0   Max.   :4.00   Max.   :0   Max.   :0
##
##              Type     Simple_UC       Average_UC      Complex_UC
##  Mobile App    :5   Min.   : 0.00   Min.   : 0.00   Min.   : 0.00
##  Mobile Game   :1   1st Qu.: 2.00   1st Qu.: 2.75   1st Qu.: 0.00
##  Mobile&Web App:4   Median : 5.00   Median : 3.00   Median : 1.00
##  web App       :2   Mean   : 8.80   Mean   : 4.15   Mean   : 2.05
##  Web App       :8   3rd Qu.:13.75   3rd Qu.: 6.00   3rd Qu.: 3.25
##                     Max.   :28.00   Max.   :10.00   Max.   :10.00
##
##  Normalized_UC_Effort
##  Min.   :  8.016  
##  1st Qu.: 16.502  
##  Median : 38.315  
##  Mean   : 55.634  
##  3rd Qu.: 86.024  
##  Max.   :186.051  
## 
```

Preprocessing the data

Replacing all the NaN with the mean value.

```
data$NT = ifelse(is.na(data$NT), ave(data$NT, FUN = function(x) mean(x, na.rm = TRUE)),data$NT)
data$INT_ALY = ifelse(is.na(data$INT_ALY), ave(data$INT_ALY, FUN = function(x) mean(x, na.rm = TRUE)),da
data$INT = ifelse(is.na(data$INT), ave(data$INT, FUN = function(x) mean(x, na.rm = TRUE)),data$INT)
data$DM = ifelse(is.na(data$DM), ave(data$DM, FUN = function(x) mean(x, na.rm = TRUE)),data$DM)
data$CTRL = ifelse(is.na(data$CTRL), ave(data$CTRL, FUN = function(x) mean(x, na.rm = TRUE)),data$CTRL)
data$EXTCLL = ifelse(is.na(data$EXTCLL), ave(data$EXTCLL, FUN = function(x) mean(x, na.rm = TRUE)),data$
data$EXTIVK = ifelse(is.na(data$EXTIVK), ave(data$EXTIVK, FUN = function(x) mean(x, na.rm = TRUE)),data$
```

Preparing the independent variables

1. Removing all the variables with zero value for all the observations.
2. Facorizing the type variable
3. Calculating the corelation between all the independent and dependent variables.
4. Choosing all the variables with highest corelation values.

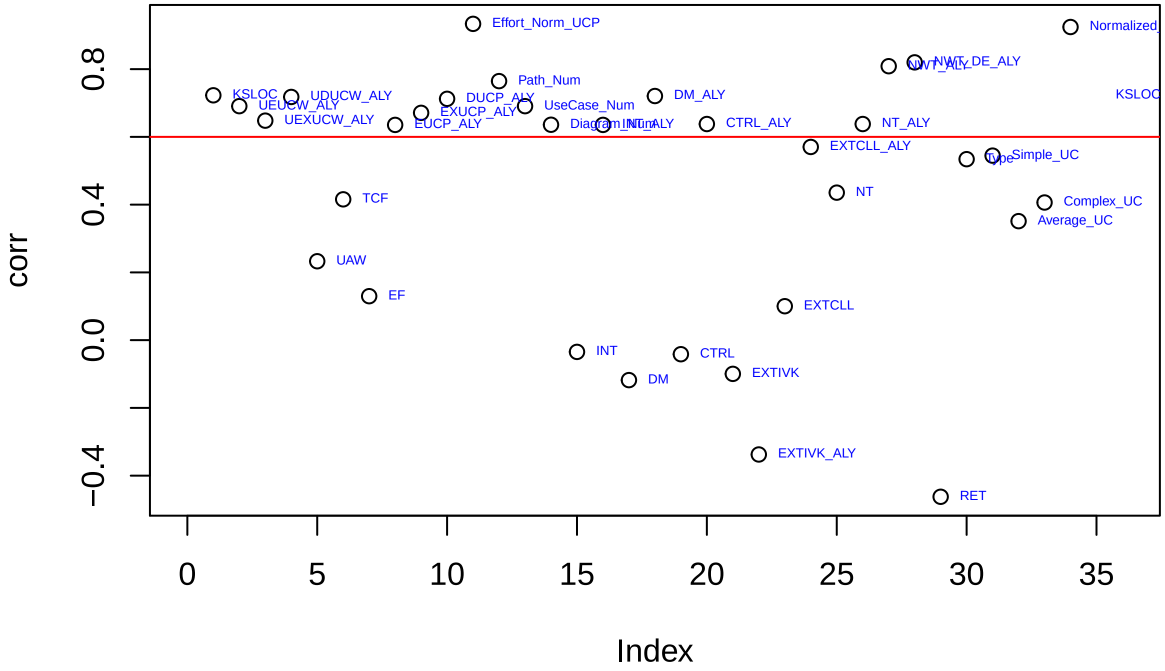
```
x <-data[,7:45];
x$Type = factor(x$Type, levels = c('Web App', 'Mobile App', 'Mobile&Web App'),labels = c(1,2,3))
x$ILF<-NULL
x$EIF<-NULL
x$DET<-NULL
x$EF_ALY<-NULL
x$TCF_ALY<-NULL
x$Type[5] = 1
x$Type[7] = 1
x$Type[20] = 3
x$Type = as.numeric(x$Type)

y =data$Effort
summary(x)
```

```
##      KSLOC          UEUCW_ALY       UEXUCW_ALY        UDUCW_ALY     
##  Min.   : 0.552   Min.   : 75.0   Min.   : 42.00   Min.   : 29.00  
##  1st Qu.: 2.777   1st Qu.:120.0   1st Qu.: 96.75   1st Qu.: 87.75  
##  Median : 4.920   Median :157.5   Median :174.50   Median :148.50  
##  Mean   : 5.507   Mean   :225.0   Mean   :193.10   Mean   :191.70  
##  3rd Qu.: 7.422   3rd Qu.:292.5   3rd Qu.:250.50   3rd Qu.:236.50  
##  Max.   :21.344   Max.   :705.0   Max.   :701.00   Max.   :777.00  
##       UAW            TCF               EF             EUCP_ALY     
##  Min.   : 3.00   Min.   :0.7950   Min.   :0.8750   Min.   :  77.92  
##  1st Qu.: 6.00   1st Qu.:0.8738   1st Qu.:0.9463   1st Qu.: 117.55  
##  Median : 9.00   Median :0.9200   Median :1.0250   Median : 151.75  
##  Mean   : 8.55   Mean   :0.9280   Mean   :1.0265   Mean   : 243.80
##  3rd Qu.: 9.00   3rd Qu.:0.9363   3rd Qu.:1.0288   3rd Qu.: 279.56
##  Max.   :14.00   Max.   :1.1750   Max.   :1.3250   Max.   :1067.00
##    EXUCP_ALY          DUCP_ALY       Effort_Norm_UCP     Path_Num
##  Min.   :  61.57   Min.   :  49.26   Min.   : 194.1   Min.   : 19.00
##  1st Qu.:  94.24   1st Qu.:  94.71   1st Qu.: 528.4   1st Qu.: 33.50
##  Median : 161.15   Median : 135.31   Median : 681.4   Median : 53.00
##  Mean   : 200.49   Mean   : 201.15   Mean   :1165.7   Mean   : 62.55
##  3rd Qu.: 215.71   3rd Qu.: 206.74   3rd Qu.:1745.5   3rd Qu.: 76.00
##  Max.   :1061.06   Max.   :1173.84   Max.   :3265.0   Max.   :246.00
##   UseCase_Num    Diagram_Num        INT           INT_ALY
##  Min.   : 5.0   Min.   : 5.0   Min.   : 1.00   Min.   :  2.00
##  1st Qu.: 8.0   1st Qu.:10.0   1st Qu.: 2.00   1st Qu.: 13.75
##  Median :10.5   Median :12.0   Median : 6.50   Median : 25.50
##  Mean   :15.0   Mean   :16.7   Mean   : 6.70   Mean   : 29.50
##  3rd Qu.:19.5   3rd Qu.:21.0   3rd Qu.:10.25   3rd Qu.: 36.25
##  Max.   :47.0   Max.   :47.0   Max.   :15.00   Max.   :119.00
##        DM            DM_ALY           CTRL          CTRL_ALY
##  Min.   : 1.00   Min.   : 8.00   Min.   : 1.00   Min.   : 17.00
##  1st Qu.: 2.00   1st Qu.:12.50   1st Qu.: 2.75   1st Qu.: 28.50
##  Median : 5.00   Median :17.50   Median : 7.50   Median : 49.50
##  Mean   : 5.40   Mean   :23.15   Mean   : 7.40   Mean   : 52.85
##  3rd Qu.: 8.25   3rd Qu.:30.75   3rd Qu.:11.25   3rd Qu.: 71.25
##  Max.   :12.00   Max.   :57.00   Max.   :16.00   Max.   :168.00
##      EXTIVK       EXTIVK_ALY       EXTCLL      EXTCLL_ALY
##  Min.   :1.00   Min.   :0.00   Min.   :1.0   Min.   : 0.00
##  1st Qu.:1.00   1st Qu.:0.00   1st Qu.:1.0   1st Qu.: 0.00
##  Median :1.00   Median :1.00   Median :1.0   Median : 0.00
##  Mean   :2.25   Mean   :1.50   Mean   :1.7   Mean   : 1.35
##  3rd Qu.:4.00   3rd Qu.:2.25   3rd Qu.:2.0   3rd Qu.: 2.00
##  Max.   :6.00   Max.   :6.00   Max.   :5.0   Max.   :11.00
##        NT            NT_ALY          NWT_ALY         NWT_DE_ALY
##  Min.   : 1.00   Min.   : 17.00   Min.   : 118.0   Min.   : 116.0
##  1st Qu.: 5.00   1st Qu.: 28.50   1st Qu.: 232.5   1st Qu.: 233.5
##  Median : 8.50   Median : 49.50   Median : 329.0   Median : 328.0
##  Mean   : 8.60   Mean   : 52.85   Mean   : 520.5   Mean   : 536.8
##  3rd Qu.:13.25   3rd Qu.: 71.25   3rd Qu.: 570.2   3rd Qu.: 581.5
##  Max.   :14.00   Max.   :168.00   Max.   :2332.0   Max.   :2435.0
##       RET            Type        Simple_UC       Average_UC
##  Min.   :0.00   Min.   :1.00   Min.   : 0.00   Min.   : 0.00
##  1st Qu.:0.00   1st Qu.:1.00   1st Qu.: 2.00   1st Qu.: 2.75
##  Median :2.00   Median :1.50   Median : 5.00   Median : 3.00
##  Mean   :1.65   Mean   :1.75   Mean   : 8.80   Mean   : 4.15
##  3rd Qu.:2.00   3rd Qu.:2.25   3rd Qu.:13.75   3rd Qu.: 6.00
##  Max.   :4.00   Max.   :3.00   Max.   :28.00   Max.   :10.00
##    Complex_UC    Normalized_UC_Effort
##  Min.   : 0.00   Min.   :  8.016
##  1st Qu.: 0.00   1st Qu.: 16.502
##  Median : 1.00   Median : 38.315
##  Mean   : 2.05   Mean   : 55.634
##  3rd Qu.: 3.25   3rd Qu.: 86.024
##  Max.   :10.00   Max.   :186.051
```

Correlation

Calculating the correlation and choosing the independent variables with correlation higher than 0.6 with the dependent variable (Effort).

```
corr <- cor(x,y)
plot(corr,xlim=c(0, 36))
text(1:35,corr,row.names(corr),cex=0.4, pos=4, col="blue")
abline(h=0.6,col="red")
```



Looking at the graph, following are the most correlated independent variables:
1. KSLOC
2. UEUCW_ALY
3. UEXUCW_ALY
4. UDUCW_ALY
5. Effort_Norm_UCP
6. Path_Num
7. DUCP_ALY
8. EXUCP_ALY
9. EUCP_ALY
10. UseCase_Num
11. Diagram_Num
12. INT_ALY
13. DM_ALY
14. CTRL_ALY
15. NT_ALY
16. NWT_DE_ALY
17. NWT_ALY

Model Fitting

Using all the above variables except UseCase_NUM and Diagram_Num for fitting the model.

```r
independentVar <- data.frame(x$KSLOC,x$UEUCW_ALY,x$UEXUCW_ALY,x$UDUCW_ALY,x$Effort_Norm_UCP,x$Path_Num,

names(independentVar)<- c("KSLOC","UEUCW_ALY","UEXUCW_ALY","UDUCW_ALY","Effort_Norm_UCP","Path_Num","DU

#library(caret)
#set.seed(30)
#model <- train(y~.,data=independentVar,method="lm",trControl = trainControl(method = "cv", number=2,ve

fit <- lm(y~.,data=independentVar)
summary(fit)
```

```
##
## Call:
## lm(formula = y ~ ., data = independentVar)
##
## Residuals:
##         1         2         3         4         5         6         7
##  167.0279 -194.0992 -171.8621    8.9569   -0.2924 -118.3572  138.1005
##         8         9        10        11        12        13        14
##   68.4321  185.5159   89.8604 -108.4177 -146.6986   -5.9901    0.8631
##        15        16        17        18        19        20
##    2.8460   27.3099   28.1691   27.0683    6.7885   -5.2211
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     183.3461   312.2033   0.587   0.5825
## KSLOC            33.5568    38.4788   0.872   0.4231
## UEUCW_ALY        -1.9482     3.2776  -0.594   0.5781
## UEXUCW_ALY       48.1700    35.3833   1.361   0.2315
## UDUCW_ALY       -45.8973    43.8570  -1.047   0.3433
## Effort_Norm_UCP   0.3988     0.1180   3.380   0.0197 *
## Path_Num         -3.0924    43.3269  -0.071   0.9459
## DUCP_ALY         43.0490    58.1155   0.741   0.4921
## EXUCP_ALY       -49.4032    65.3425  -0.756   0.4837
## EUCP_ALY          2.5930     3.4164   0.759   0.4821
## INT_ALY          84.4573   128.2986   0.658   0.5395
## DM_ALY           77.0080    39.9469   1.928   0.1118
## CTRL_ALY        -68.2261    58.8718  -1.159   0.2988
## NWT_DE_ALY        9.6134     5.2364   1.836   0.1258
## NWT_ALY          -9.3280     8.1313  -1.147   0.3032
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 205.5 on 5 degrees of freedom
## Multiple R-squared:  0.988,  Adjusted R-squared:  0.9544
## F-statistic: 29.43 on 14 and 5 DF,  p-value: 0.0007453
```

```r
raw_data <- read.csv(file = "/Users/mohit/Development/My Scripts/modelsEvaluation.csv", stringsAsFactor
```

```r
# if there is some missing value
sort(sapply(raw_data, function(x) {
            sum(is.na(x))
}), decreasing  = T)
```

```
##                   NUM                 PROJ               Effort
##                    0                    0                    0
##           Effort_ALY          Effort_Norm          Norm_Factor
##                    0                    0                    0
##                KSLOC            UEUCW_ALY           UEXUCW_ALY
##                    0                    0                    0
##            UDUCW_ALY                  UAW                  TCF
##                    0                    0                    0
##              TCF_ALY                   EF               EF_ALY
##                    0                    0                    0
##             EUCP_ALY            EXUCP_ALY             DUCP_ALY
##                    0                    0                    0
##      Effort_Norm_UCP             Path_Num          UseCase_Num
##                    0                    0                    0
##          Diagram_Num                  INT              INT_ALY
##                    0                    0                    0
##                   DM               DM_ALY                 CTRL
##                    0                    0                    0
##             CTRL_ALY               EXTIVK           EXTIVK_ALY
##                    0                    0                    0
##               EXTCLL           EXTCLL_ALY                   NT
##                    0                    0                    0
##               NT_ALY              NWT_ALY           NWT_DE_ALY
##                    0                    0                    0
##                  DET                  RET                  ILF
##                    0                    0                    0
##                  EIF                 Type            Simple_UC
##                    0                    0                    0
##           Average_UC           Complex_UC Normalized_UC_Effort
##                    0                    0                    0
```

```
which(sapply(raw_data, function(x){sum(x == 'undefined') > 0}))
```

```
##    INT     DM   CTRL EXTIVK EXTCLL     NT
##     23     25     27     29     31     33
```

```
raw_data[which(raw_data$INT == 'undefined'),'INT'] = 0
raw_data[which(raw_data$DM == 'undefined'),'DM'] = 0
raw_data[which(raw_data$CTRL == 'undefined'),'CTRL'] = 0
raw_data[which(raw_data$EXTIVK == 'undefined'),'EXTIVK'] = 0
raw_data[which(raw_data$EXTCLL == 'undefined'),'EXTCLL'] = 0
raw_data[which(raw_data$NT == 'undefined'),'NT'] = 0
raw_data[which(raw_data$NT == 'NaN'),'NT'] = 0
```

```
# check type of each column
sapply(raw_data, mode)
```

```
##                  NUM                 PROJ               Effort
##            "numeric"          "character"            "numeric"
##           Effort_ALY          Effort_Norm          Norm_Factor
##            "numeric"            "numeric"            "numeric"
##                KSLOC            UEUCW_ALY           UEXUCW_ALY
##            "numeric"            "numeric"            "numeric"
##            UDUCW_ALY                  UAW                  TCF
##            "numeric"            "numeric"            "numeric"
##              TCF_ALY                   EF               EF_ALY
##            "numeric"            "numeric"            "numeric"
##             EUCP_ALY            EXUCP_ALY             DUCP_ALY
##            "numeric"            "numeric"            "numeric"
##      Effort_Norm_UCP             Path_Num          UseCase_Num
##            "numeric"            "numeric"            "numeric"
##          Diagram_Num                  INT              INT_ALY
##            "numeric"          "character"            "numeric"
##                   DM               DM_ALY                 CTRL
##          "character"            "numeric"          "character"
##             CTRL_ALY               EXTIVK           EXTIVK_ALY
##            "numeric"          "character"            "numeric"
##               EXTCLL           EXTCLL_ALY                   NT
##          "character"            "numeric"          "character"
##               NT_ALY              NWT_ALY           NWT_DE_ALY
##            "numeric"            "numeric"            "numeric"
##                  DET                  RET                  ILF
##            "numeric"            "numeric"            "numeric"
##                  EIF                 Type            Simple_UC
##            "numeric"          "character"            "numeric"
##           Average_UC           Complex_UC Normalized_UC_Effort
##            "numeric"            "numeric"            "numeric"
```

```
# transfer type of columns
raw_data <- transform(raw_data, INT = as.numeric(INT),
          DM = as.numeric(DM),
          CTRL = as.numeric(CTRL),
          EXTIVK = as.numeric(EXTIVK),
          EXTCLL = as.numeric(EXTCLL),
          NT = as.numeric(NT),
          Type = as.factor(Type))
```

```
# check again
sapply(raw_data, mode)
```

```
##                  NUM                 PROJ               Effort
##            "numeric"          "character"            "numeric"
##           Effort_ALY          Effort_Norm          Norm_Factor
##            "numeric"            "numeric"            "numeric"
##                KSLOC            UEUCW_ALY           UEXUCW_ALY
##            "numeric"            "numeric"            "numeric"
##            UDUCW_ALY                  UAW                  TCF
##            "numeric"            "numeric"            "numeric"
##              TCF_ALY                   EF               EF_ALY
##            "numeric"            "numeric"            "numeric"
##             EUCP_ALY            EXUCP_ALY             DUCP_ALY
##            "numeric"            "numeric"            "numeric"
##      Effort_Norm_UCP             Path_Num          UseCase_Num
##            "numeric"            "numeric"            "numeric"
##          Diagram_Num                  INT              INT_ALY
##            "numeric"            "numeric"            "numeric"
##                   DM               DM_ALY                 CTRL
##            "numeric"            "numeric"            "numeric"
##             CTRL_ALY               EXTIVK           EXTIVK_ALY
##            "numeric"            "numeric"            "numeric"
##               EXTCLL           EXTCLL_ALY                   NT
##            "numeric"            "numeric"            "numeric"
##               NT_ALY              NWT_ALY           NWT_DE_ALY
##            "numeric"            "numeric"            "numeric"
##                  DET                  RET                  ILF
##            "numeric"            "numeric"            "numeric"
##                  EIF                 Type            Simple_UC
##            "numeric"            "numeric"            "numeric"
##           Average_UC           Complex_UC Normalized_UC_Effort
##            "numeric"            "numeric"            "numeric"
```

```
# X_data <- subset(raw_data, select = -c(NUM, PROJ, Effort, Effort_ALY, Effort_Norm, Norm_Factor))
X_data = subset(raw_data, select = c("EF","TCF","Type","KSLOC","Normalized_UC_Effort",
                       "UAW","Average_UC","RET","EXTIVK"))
Y_data <- raw_data[,"Effort"]
```

```
X_data[which(X_data$Type == 'Mobile App' | X_data$Type == 'Mobile Game'), 'type'] = 0
X_data[which(X_data$Type == 'Web App' | X_data$Type == 'web App'), 'type'] = 1
X_data[which(X_data$Type == 'Mobile&Web App'), 'type'] = 2
```

```
X_data = subset(X_data, select = -c(Type))
```

```
# scale numberic features
myscale = function(x) sqrt(sum((x - mean(x)) ^ 2) / length(x))
sx = as.matrix(scale(X_data, scale = apply(X_data, 2, myscale)))
sy = as.vector(scale(Y_data, scale = myscale(Y_data)))
```

```
# X_data <- model.matrix(~., X_data)
```

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loading required package: foreach
```

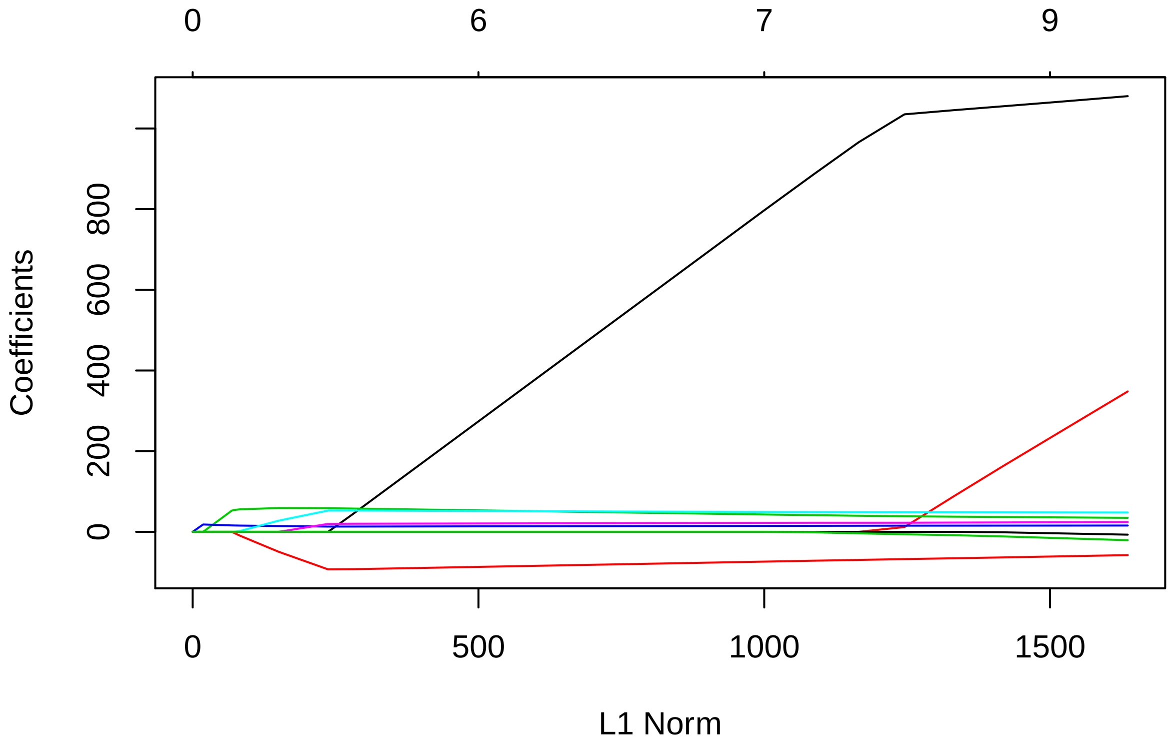
```
## Loaded glmnet 2.0-13
```

```
lasso_lm <- glmnet(x = as.matrix(X_data), y = as.vector(Y_data), alpha = 1, standardize = F)
```

```
lasso_lm$lambda
```

```
##   [1] 40228.155501 36654.397040 33398.121430 30431.124372 27727.707154
##   [6] 25264.454071 23020.029603 20974.993619 19111.632996 17413.808194
##  [11] 15866.813469 14457.249491 13172.907292 12002.662514 10936.379057
##  [16]  9964.821284  9079.574025  8272.969693  7538.021867  6868.364780
##  [21]  6258.198183  5702.237104  5195.666076  4734.097422  4313.533255
##  [26]  3930.330849  3581.171089  3263.029719  2973.151152  2709.024598
##  [31]  2468.362319  2249.079814  2049.277762  1867.225574  1701.346400
##  [36]  1550.203475  1412.487671  1287.006159  1172.672079  1068.495124
##  [41]   973.572962   887.083423   808.277377   736.472242   671.046078
##  [46]   611.432194   557.114243   507.621749   462.526032   421.436494
##  [51]   383.997237   349.883981   318.801253   290.479828   264.674401
##  [56]   241.161456   219.737337   200.216478   182.429798   166.223237
##  [61]   151.456423   138.001452   125.741783   114.571230   104.393038
##  [66]    95.119048    86.668934    78.969505    71.954071    65.561870
##  [71]    59.737534    54.430616    49.595150    45.189254    41.174765
##  [76]    37.516912    34.184013    31.147200    28.380168    25.858952
##  [81]    23.561714    21.468557    19.561349    17.823573    16.240176
##  [86]    14.797443    13.482879    12.285097    11.193723    10.199304
##  [91]     9.293226     8.467641     7.715399     7.029984     6.405460
## [96]      5.836417      5.317925      4.845496      4.415035      4.022816
```

```
plot(lasso_lm)
```



```
#library(plotmo) # for plot_glmnet
```

```
# for 10 biggest final features
#plot_glmnet(lasso_lm)                              # default colors
#plot_glmnet(lasso_lm, label=10)
```

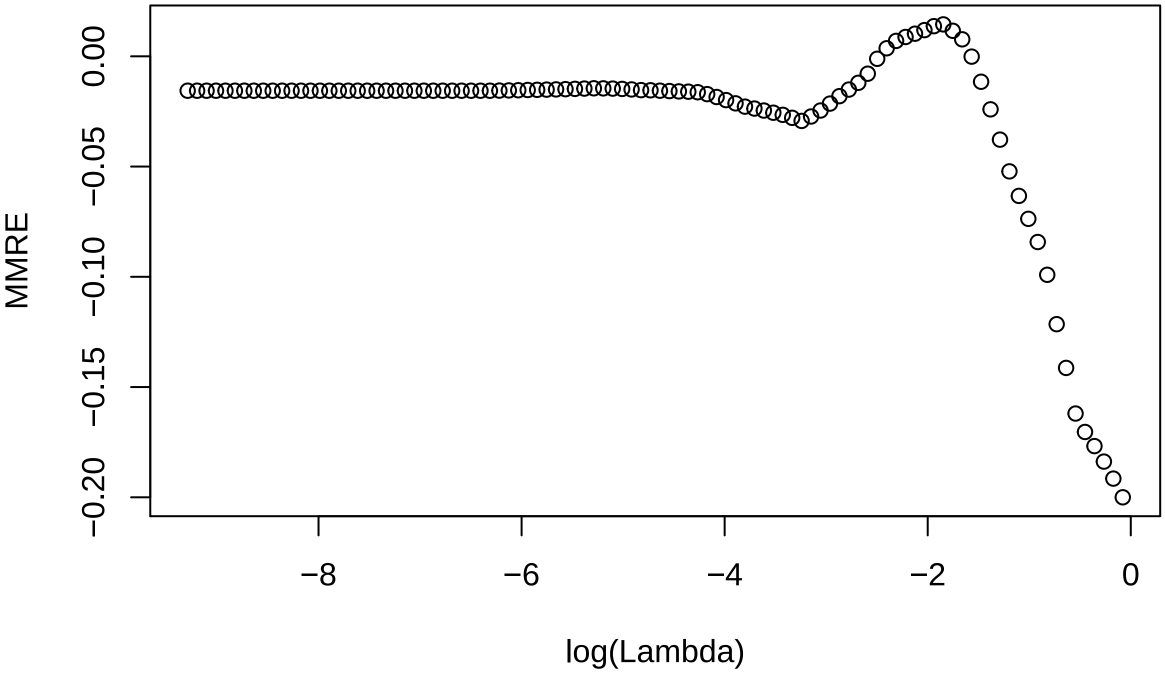
```
Lasso_range = function(x, y, k){
  # inputs:
      # x, independent variables
      # y: dependent varaibles
      # k: the length of sequence
  # output:
      # seq: a sequence of lambdaa from high to low


  # define my own scale function to simulate that in glmnet
  # myscale = function(x) sqrt(sum((x - mean(x)) ^ 2) / length(x))
  #
  # # normalize x and y
  # sx = as.matrix(scale(x, scale = apply(x, 2, myscale)))
  # sy = as.vector(scale(y, scale = myscale(y)))

  sx = as.matrix(x)
  sy = as.vector(y)

  max_lambda = max(abs(colSums(sx * sy))) / dim(x)[1]
  # The default depends on the sample size nobs relative to the number of variables nvars.
  # If nobs > nvars, the default is 0.0001, close to zero.

  # If nobs < nvars, the default is 0.01.
  # A very small value of lambda.min.ratio will lead to a saturated fit in the nobs < nvars case.
  ratio = 0
  if(dim(sx)[1] > dim(sx)[2]){
    ratio = 0.0001
  }else{
    ratio = 0.01
  }

  min_lambda = max_lambda * ratio

  log_seq = seq(from  = log(min_lambda), to = log(max_lambda), length.out = k)
  seq = sort(exp(log_seq), decreasing = T)
  return(seq)
}
```

```
Lasso_range(sx, sy, 100)
```

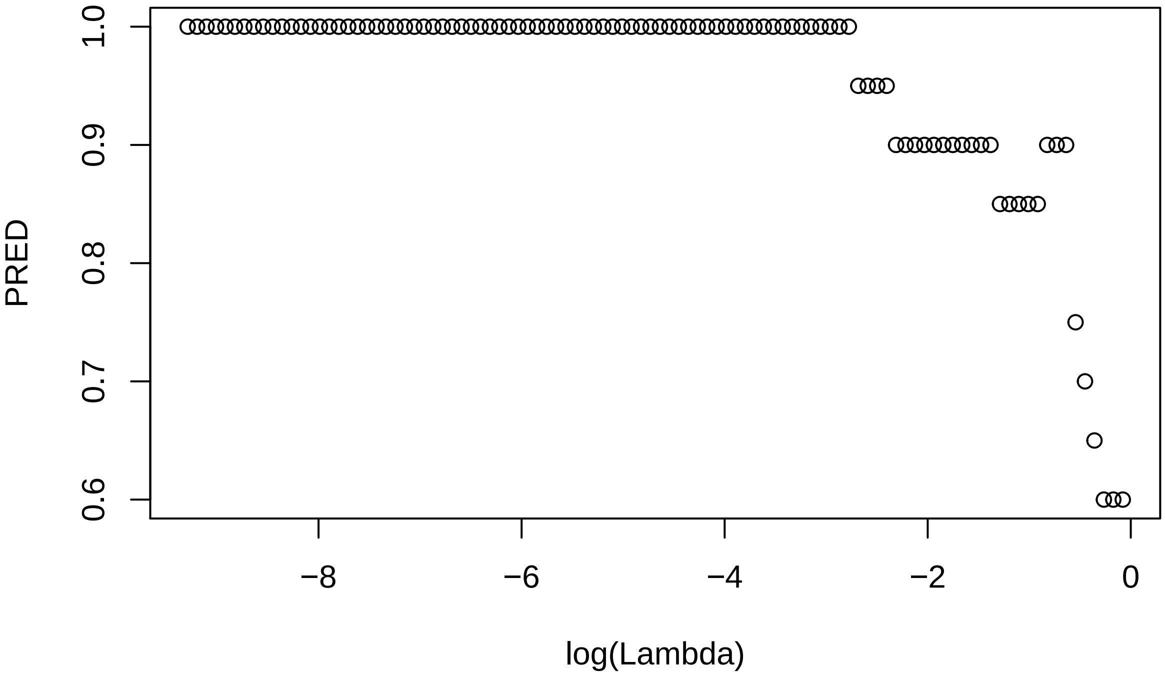
```
##   [1] 9.243772e-01 8.422581e-01 7.674342e-01 6.992574e-01 6.371373e-01
##   [6] 5.805358e-01 5.289626e-01 4.819710e-01 4.391540e-01 4.001408e-01
##  [11] 3.645934e-01 3.322039e-01 3.026918e-01 2.758015e-01 2.513001e-01
##  [16] 2.289753e-01 2.086338e-01 1.900993e-01 1.732114e-01 1.578238e-01
##  [21] 1.438032e-01 1.310281e-01 1.193879e-01 1.087818e-01 9.911793e-02
##  [26] 9.031257e-02 8.228945e-02 7.497908e-02 6.831815e-02 6.224895e-02
##  [31] 5.671893e-02 5.168017e-02 4.708905e-02 4.290579e-02 3.909416e-02
##  [36] 3.562114e-02 3.245665e-02 2.957330e-02 2.694609e-02 2.455227e-02
##  [41] 2.237111e-02 2.038373e-02 1.857289e-02 1.692293e-02 1.541954e-02
##  [46] 1.404971e-02 1.280157e-02 1.166432e-02 1.062809e-02 9.683921e-03
##  [51] 8.823628e-03 8.039761e-03 7.325531e-03 6.674751e-03 6.081785e-03
##  [56] 5.541496e-03 5.049204e-03 4.600647e-03 4.191938e-03 3.819538e-03
##  [61] 3.480221e-03 3.171047e-03 2.889340e-03 2.632659e-03 2.398781e-03
##  [66] 2.185680e-03 1.991510e-03 1.814590e-03 1.653387e-03 1.506504e-03
##  [71] 1.372671e-03 1.250726e-03 1.139615e-03 1.038375e-03 9.461287e-04
##  [76] 8.620772e-04 7.854927e-04 7.157117e-04 6.521298e-04 5.941964e-04
##  [81] 5.414096e-04 4.933123e-04 4.494878e-04 4.095565e-04 3.731727e-04
##  [86] 3.400210e-04 3.098145e-04 2.822914e-04 2.572134e-04 2.343633e-04
##  [91] 2.135431e-04 1.945725e-04 1.772872e-04 1.615375e-04 1.471870e-04
##  [96] 1.341113e-04 1.221972e-04 1.113416e-04 1.014503e-04 9.243772e-05
```

```
set.seed(2)
lambda_list <- Lasso_range(sx,sy,100)
percent = 50
cvfit = cv.glmnet(data.matrix(sx),sy,
                  standardize = F, type.measure = 'mse', nfolds = 5, alpha = 1)
# # 5 fold cross validation
 k <- 5
#
# function to calculate MMRE
calcMMRE <- function(testData,pred){
  mmre <- abs(testData - pred)/testData
  mean_value <- mean(mmre)
  mean_value
}
# # function to calculate PRED
calcPRED <- function(testData,pred,percent){
  value <- abs(testData - pred)/testData
  percent_value <- percent/100
  pred_value <- value <= percent_value
  mean(pred_value)
}
#
 folds <- cut(seq(1,nrow(sx)),breaks=k,labels=FALSE)
 mean_mmre <- vector("list",k)
 mean_pred <- vector("list",k)
 overall_mean_mmre <- vector("list",100)
 overall_mean_pred <- vector("list",100)
 for(iterator in seq(1,100)){
   for(i in 1:k){
     testIndexes <- which(folds==i,arr.ind=TRUE)
     testData <- sy[testIndexes]
     trainData <- sx[-testIndexes,]
     pred <- predict(cvfit,newx=data.matrix(sx),s=lambda_list[[iterator]])
     #print(paste("Iterator",iterator, i),sep=" ")
     mean_mmre[[i]] <- calcMMRE(testData,pred[testIndexes])
     mean_pred[[i]] <- calcPRED(testData,pred[testIndexes],percent)
 }
 overall_mean_mmre[[iterator]] <- mean(as.numeric(mean_mmre))
 overall_mean_pred[[iterator]] <- mean(as.numeric(mean_pred))
 #print(overall_mean_mmre[[iterator]])
 #print(overall_mean_pred[[iterator]])
 }
```

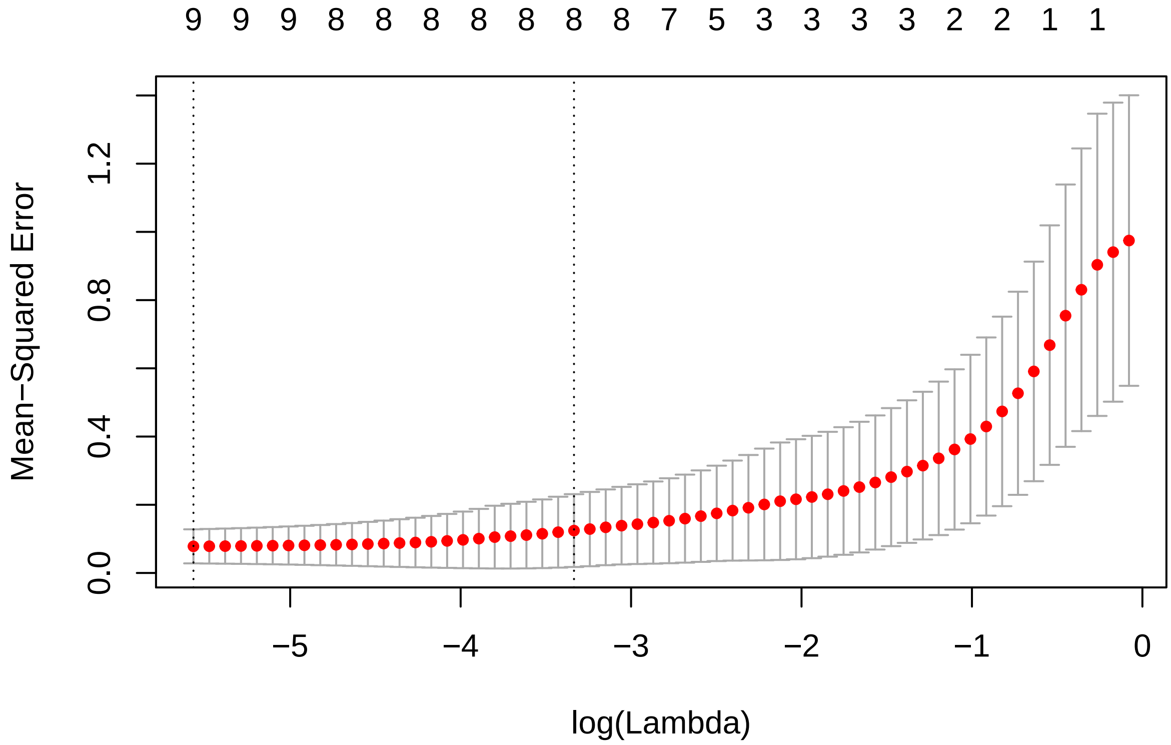
```
plot(log(lambda_list),overall_mean_mmre,xlab="log(Lambda)",ylab="MMRE")
```



```r
plot(log(lambda_list),overall_mean_pred,xlab="log(Lambda)",ylab = "PRED")
```



```r
plot(cvfit)
```


Mean−Squared Error
log(Lambda)