Report

Atharva and Mohit

April 25, 2018

Loading data form the csv file

```
data <- read.csv('/Users/mohit/Development/My Scripts/modelEvaluation_4_20_cleaned_1.csv',stringsAsFact
summary(data)
```

```
##       NUM
##  Min.   : 1.000
##  1st Qu.: 3.000
##  Median : 7.000
##  Mean   : 7.548
##  3rd Qu.:11.500
##  Max.   :17.000
##
##                                                                    PROJ
##  F15a_construction_meeting_minutes_application                       : 2
##  F13a_LA_Commons_upgradeof_website                                   : 1
##  F13a_LiveRiot_Video_Editing_System_and_socialNetworking_enhancement: 1
##  F13a_OnlineWedding_Management_System                                : 1
##  F13a_Surgery_Assist                                                 : 1
##  F13a_Yanomamo Interactive CDROM                                     : 1
##  (Other)                                                             :24
##   Effort_Norm      Norm_Factor         KSLOC            Effort
##  Min.   : 135.5   Min.   :0.6694   Min.   : 0.552   Min.   : 103.0
##  1st Qu.: 468.6   1st Qu.:0.9747   1st Qu.: 2.228   1st Qu.: 285.5
##  Median : 781.8   Median :1.1927   Median : 4.402   Median : 759.0
##  Mean   :1465.0   Mean   :1.2523   Mean   : 4.757   Mean   :1139.1
##  3rd Qu.:2431.6   3rd Qu.:1.4471   3rd Qu.: 7.336   3rd Qu.:1392.9
##  Max.   :5850.4   Max.   :1.9079   Max.   :12.263   Max.   :8224.7
##  NA's   :8        NA's   :8        NA's   :1
##  Effort_Norm_UCP     Path_Num      UseCase_Num    Total_Degree
##  Min.   : 140.6   Min.   : 27.0   Min.   : 5.0   Min.   :0
##  1st Qu.: 344.1   1st Qu.: 55.0   1st Qu.: 8.0   1st Qu.:0
##  Median : 652.0   Median : 91.0   Median :11.0   Median :0
##  Mean   :1008.1   Mean   :129.0   Mean   :13.1   Mean   :0
##  3rd Qu.:1546.8   3rd Qu.:143.5   3rd Qu.:17.0   3rd Qu.:0
##  Max.   :3217.0   Max.   :488.0   Max.   :26.0   Max.   :0
##  NA's   :8
##   Element_Num      Entity_Num    attribute_num    operation_num
##  Min.   : 59.0   Min.   : 0.00   Min.   :  0.00   Min.   :  0.00
##  1st Qu.:128.0   1st Qu.: 9.00   1st Qu.:  0.00   1st Qu.:  0.00
##  Median :183.0   Median :11.00   Median : 13.00   Median :  0.00
##  Mean   :209.6   Mean   :15.42   Mean   : 39.26   Mean   : 18.19
##  3rd Qu.:250.0   3rd Qu.:17.50   3rd Qu.: 57.50   3rd Qu.: 14.00
##  Max.   :555.0   Max.   :49.00   Max.   :221.00   Max.   :197.00
##
##    class_num     Top_Level_Classes Average_Depth_Inheritance_Tree
##  Min.   : 0.00   Min.   :  0.0     Min.   :0.00000
##  1st Qu.: 9.00   1st Qu.: 12.5     1st Qu.:0.00000
##  Median :11.00   Median : 20.0     Median :0.00000
##  Mean   :15.42   Mean   : 66.0     Mean   :0.04285
##  3rd Qu.:17.50   3rd Qu.: 49.5     3rd Qu.:0.09601
##  Max.   :49.00   Max.   :288.0     Max.   :0.22414
##
##  Average_Number_Of_Children_Per_Base_Class
##  Min.   :0.00000
##  1st Qu.:0.00000
##  Median :0.00000
##  Mean   :0.04236
##  3rd Qu.:0.06452
##  Max.   :0.44000
##
##  Number_Of_Inheritance_Relationships Number_Of_Derived_Classes
##  Min.   : 0.000                      Min.   : 0.000
##  1st Qu.: 0.000                      1st Qu.: 0.000
##  Median : 0.000                      Median : 0.000
##  Mean   : 4.484                      Mean   : 3.774
##  3rd Qu.: 2.000                      3rd Qu.: 1.500
##  Max.   :24.000                      Max.   :22.000
##
##  Number_Of_Classes_Inherited Number_Of_Classes_Inherited_From
##  Min.   : 0.000              Min.   : 0.000
##  1st Qu.: 0.000              1st Qu.: 0.000
##  Median : 0.000              Median : 0.000
##  Mean   : 4.484              Mean   : 6.484
##  3rd Qu.: 2.000              3rd Qu.: 2.000
##  Max.   :24.000              Max.   :60.000
##
##  Number_Of_Children Depth_Inheritance_Tree Coupling_Between_Objects
##  Min.   : 0.000     Min.   : 0.000         Min.   : 0.000
##  1st Qu.: 0.000     1st Qu.: 0.000         1st Qu.: 0.000
##  Median : 0.000     Median : 0.000         Median : 0.000
##  Mean   : 3.774     Mean   : 5.258         Mean   : 6.484
##  3rd Qu.: 1.500     3rd Qu.: 2.000         3rd Qu.: 2.000
##  Max.   :22.000     Max.   :28.000         Max.   :60.000
##
##     para_num        usage_num    real_num   assoc_num externaloper_num
##  Min.   :  0.00   Min.   :0   Min.   :0   Min.   :0   Min.   :  0.00
##  1st Qu.:  0.00   1st Qu.:0   1st Qu.:0   1st Qu.:0   1st Qu.:  0.00
##  Median :  0.00   Median :0   Median :0   Median :0   Median :  0.00
##  Mean   : 18.06   Mean   :0   Mean   :0   Mean   :0   Mean   : 18.19
##  3rd Qu.:  3.00   3rd Qu.:0   3rd Qu.:0   3rd Qu.:0   3rd Qu.: 14.00
##  Max.   :197.00   Max.   :0   Max.   :0   Max.   :0   Max.   :197.00
##
##  objectdata_num   avg_operation    avg_attribute    avg_parameter
##  Min.   : 0.000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.: 0.000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median : 4.000   Median :0.0000   Median :0.9091   Median :0.0000
##  Mean   : 5.323   Mean   :0.6370   Mean   :1.6694   Mean   :0.5182
##  3rd Qu.: 6.000   3rd Qu.:0.8718   3rd Qu.:3.1553   3rd Qu.:0.1364
##  Max.   :33.000   Max.   :5.7941   Max.   :5.3333   Max.   :5.7941
##
##   avg_instVar        FUNC_NA             EI              INT
##  Min.   :0.0000   Min.   : 1.000   Min.   : 0.000   Min.   : 0.00
##  1st Qu.:0.0000   1st Qu.: 5.000   1st Qu.: 0.000   1st Qu.: 9.00
##  Median :0.9091   Median : 8.000   Median : 0.000   Median :17.00
##  Mean   :1.6694   Mean   : 9.452   Mean   : 1.387   Mean   :21.45
##  3rd Qu.:3.1553   3rd Qu.:10.500   3rd Qu.: 0.000   3rd Qu.:26.00
##  Max.   :5.3333   Max.   :39.000   Max.   :30.000   Max.   :79.00
##
##        DM              CTRL            EXTCLL         TRAN_NA
##  Min.   : 0.000   Min.   :  1.00   Min.   : 0.00   Min.   :  0.0
##  1st Qu.: 0.000   1st Qu.: 14.00   1st Qu.: 9.00   1st Qu.: 35.0
##  Median : 0.000   Median : 29.00   Median :18.00   Median : 56.0
##  Mean   : 1.387   Mean   : 32.87   Mean   :22.84   Mean   : 96.1
##  3rd Qu.: 0.000   3rd Qu.: 41.00   3rd Qu.:29.00   3rd Qu.:109.5
##  Max.   :30.000   Max.   :100.00   Max.   :79.00   Max.   :423.0
##
##        NT           Complex_UC       UEUCW           UEXUCW
##  Min.   :  1.00   Min.   : 5.0   Min.   : 75.0   Min.   :  2.00
##  1st Qu.: 14.00   1st Qu.: 8.0   1st Qu.:120.0   1st Qu.: 28.00
##  Median : 29.00   Median :11.0   Median :165.0   Median : 58.00
##  Mean   : 32.87   Mean   :13.1   Mean   :196.5   Mean   : 67.68
##  3rd Qu.: 41.00   3rd Qu.:17.0   3rd Qu.:255.0   3rd Qu.: 86.50
##  Max.   :100.00   Max.   :26.0   Max.   :390.0   Max.   :227.00
##
##      UDUCW             UAW              TCF               EF
##  Min.   :  2.00   Min.   : 0.000   Min.   :0.7950   Min.   :0.8000
##  1st Qu.: 28.00   1st Qu.: 3.000   1st Qu.:0.8875   1st Qu.:0.9575
##  Median : 58.00   Median : 6.000   Median :0.9250   Median :1.0000
##  Mean   : 67.68   Mean   : 6.323   Mean   :0.9303   Mean   :0.9953
##  3rd Qu.: 86.50   3rd Qu.: 9.000   3rd Qu.:1.0000   3rd Qu.:1.0250
##  Max.   :227.00   Max.   :12.000   Max.   :1.1350   Max.   :1.2500
##
##       EUCP            EXUCP             DUCP             SWTI
##  Min.   : 77.92   Min.   :  9.64   Min.   :  9.64   Min.   : 270
##  1st Qu.:127.55   1st Qu.: 31.21   1st Qu.: 31.21   1st Qu.: 550
##  Median :165.00   Median : 56.93   Median : 56.93   Median : 910
##  Mean   :185.46   Mean   : 66.57   Mean   : 66.57   Mean   :1290
##  3rd Qu.:234.84   3rd Qu.: 86.61   3rd Qu.: 86.61   3rd Qu.:1435
##  Max.   :359.96   Max.   :191.64   Max.   :191.64   Max.   :4880
##
##      SWTII          SWTIII
##  Min.   : 290   Min.   : 214
##  1st Qu.: 825   1st Qu.: 650
##  Median :1244   Median : 982
##  Mean   :1851   Mean   :1473
##  3rd Qu.:2145   3rd Qu.:1716
##  Max.   :7234   Max.   :5780
##
```

Preprocessing the data

Replacing all the NaN with the mean value.

```
data$NUM = ifelse(is.na(data$NUM), ave(data$NUM, FUN = function(x) mean(x, na.rm = TRUE)),data$NUM)
data$PROJ = ifelse(is.na(data$PROJ), ave(data$PROJ, FUN = function(x) mean(x, na.rm = TRUE)),data$PROJ)
data$Effort_Norm = ifelse(is.na(data$Effort_Norm), ave(data$Effort_Norm, FUN = function(x) mean(x, na.r
data$Norm_Factor = ifelse(is.na(data$Norm_Factor), ave(data$Norm_Factor, FUN = function(x) mean(x, na.r
data$KSLOC = ifelse(is.na(data$KSLOC), ave(data$KSLOC, FUN = function(x) mean(x, na.rm = TRUE)),data$KS
data$Effort = ifelse(is.na(data$Effort), ave(data$Effort, FUN = function(x) mean(x, na.rm = TRUE)),data
data$Effort_Norm_UCP = ifelse(is.na(data$Effort_Norm_UCP), ave(data$Effort_Norm_UCP, FUN = function(x)
data$Path_Num = ifelse(is.na(data$Path_Num), ave(data$Path_Num, FUN = function(x) mean(x, na.rm = TRUE)
data$UseCase_Num = ifelse(is.na(data$UseCase_Num), ave(data$UseCase_Num, FUN = function(x) mean(x, na.r
data$Total_Degree = ifelse(is.na(data$Total_Degree), ave(data$Total_Degree, FUN = function(x) mean(x, na
data$Element_Num = ifelse(is.na(data$Element_Num), ave(data$Element_Num, FUN = function(x) mean(x, na.r
data$Entity_Num = ifelse(is.na(data$Entity_Num), ave(data$Entity_Num, FUN = function(x) mean(x, na.rm =
data$attribute_num = ifelse(is.na(data$attribute_num), ave(data$attribute_num, FUN = function(x) mean(x
data$operation_num = ifelse(is.na(data$operation_num), ave(data$operation_num, FUN = function(x) mean(x
data$class_num = ifelse(is.na(data$class_num), ave(data$class_num, FUN = function(x) mean(x, na.rm = TR
data$Top_Level_Classes = ifelse(is.na(data$Top_Level_Classes), ave(data$Top_Level_Classes, FUN = functi
data$Average_Depth_Inheritance_Tree = ifelse(is.na(data$Average_Depth_Inheritance_Tree), ave(data$Avera
data$Average_Number_Of_Children_Per_Base_Class = ifelse(is.na(data$Average_Number_Of_Children_Per_Base_
data$Number_Of_Inheritance_Relationships = ifelse(is.na(data$Number_Of_Inheritance_Relationships), ave(
data$Number_Of_Derived_Classes = ifelse(is.na(data$Number_Of_Derived_Classes), ave(data$Number_Of_Deriv
data$Number_Of_Classes_Inherited = ifelse(is.na(data$Number_Of_Classes_Inherited), ave(data$Number_Of_C
data$Number_Of_Classes_Inherited_From = ifelse(is.na(data$Number_Of_Classes_Inherited_From), ave(data$N
data$Number_Of_Children = ifelse(is.na(data$Number_Of_Children), ave(data$Number_Of_Children, FUN = fun
data$Depth_Inheritance_Tree = ifelse(is.na(data$Depth_Inheritance_Tree), ave(data$Depth_Inheritance_Tre
data$Coupling_Between_Objects = ifelse(is.na(data$Coupling_Between_Objects), ave(data$Coupling_Between_
data$para_num = ifelse(is.na(data$para_num), ave(data$para_num, FUN = function(x) mean(x, na.rm = TRUE)
data$usage_num = ifelse(is.na(data$usage_num), ave(data$usage_num, FUN = function(x) mean(x, na.rm = TR
data$real_num = ifelse(is.na(data$real_num), ave(data$real_num, FUN = function(x) mean(x, na.rm = TRUE)
data$assoc_num = ifelse(is.na(data$assoc_num), ave(data$assoc_num, FUN = function(x) mean(x, na.rm = TR
data$externaloper_num = ifelse(is.na(data$externaloper_num), ave(data$externaloper_num, FUN = function(
data$objectdata_num = ifelse(is.na(data$objectdata_num), ave(data$objectdata_num, FUN = function(x) mea
data$avg_operation = ifelse(is.na(data$avg_operation), ave(data$avg_operation, FUN = function(x) mean(x
data$avg_attribute = ifelse(is.na(data$avg_attribute), ave(data$avg_attribute, FUN = function(x) mean(x
data$avg_parameter = ifelse(is.na(data$avg_parameter), ave(data$avg_parameter, FUN = function(x) mean(x
data$avg_instVar = ifelse(is.na(data$avg_instVar), ave(data$avg_instVar, FUN = function(x) mean(x, na.r
data$FUNC_NA = ifelse(is.na(data$FUNC_NA), ave(data$FUNC_NA, FUN = function(x) mean(x, na.rm = TRUE)),d
data$EI = ifelse(is.na(data$EI), ave(data$EI, FUN = function(x) mean(x, na.rm = TRUE)),data$EI)
data$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$TRAN_NA = ifelse(is.na(data$TRAN_NA), ave(data$TRAN_NA, FUN = function(x) mean(x, na.rm = TRUE)),d
data$NT = ifelse(is.na(data$NT), ave(data$NT, FUN = function(x) mean(x, na.rm = TRUE)),data$NT)
data$Complex_UC = ifelse(is.na(data$Complex_UC), ave(data$Complex_UC, FUN = function(x) mean(x, na.rm =
data$UEUCW = ifelse(is.na(data$UEUCW), ave(data$UEUCW, FUN = function(x) mean(x, na.rm = TRUE)),data$UE
data$UEXUCW = ifelse(is.na(data$UEXUCW), ave(data$UEXUCW, FUN = function(x) mean(x, na.rm = TRUE)),data
data$UDUCW = ifelse(is.na(data$UDUCW), ave(data$UDUCW, FUN = function(x) mean(x, na.rm = TRUE)),data$UD
data$UAW = ifelse(is.na(data$UAW), ave(data$UAW, FUN = function(x) mean(x, na.rm = TRUE)),data$UAW)
data$TCF = ifelse(is.na(data$TCF), ave(data$TCF, FUN = function(x) mean(x, na.rm = TRUE)),data$TCF)
data$EF = ifelse(is.na(data$EF), ave(data$EF, FUN = function(x) mean(x, na.rm = TRUE)),data$EF)
data$EUCP = ifelse(is.na(data$EUCP), ave(data$EUCP, FUN = function(x) mean(x, na.rm = TRUE)),data$EUCP)
data$EXUCP = ifelse(is.na(data$EXUCP), ave(data$EXUCP, FUN = function(x) mean(x, na.rm = TRUE)),data$EX
data$DUCP = ifelse(is.na(data$DUCP), ave(data$DUCP, FUN = function(x) mean(x, na.rm = TRUE)),data$DUCP)
```

```
data$SWTI = ifelse(is.na(data$SWTI), ave(data$SWTI, FUN = function(x) mean(x, na.rm = TRUE)),data$SWTI)
data$SWTII = ifelse(is.na(data$SWTII), ave(data$SWTII, FUN = function(x) mean(x, na.rm = TRUE)),data$SW
data$SWTIII = ifelse(is.na(data$SWTIII), ave(data$SWTIII, FUN = function(x) mean(x, na.rm = TRUE)),data$
```

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.

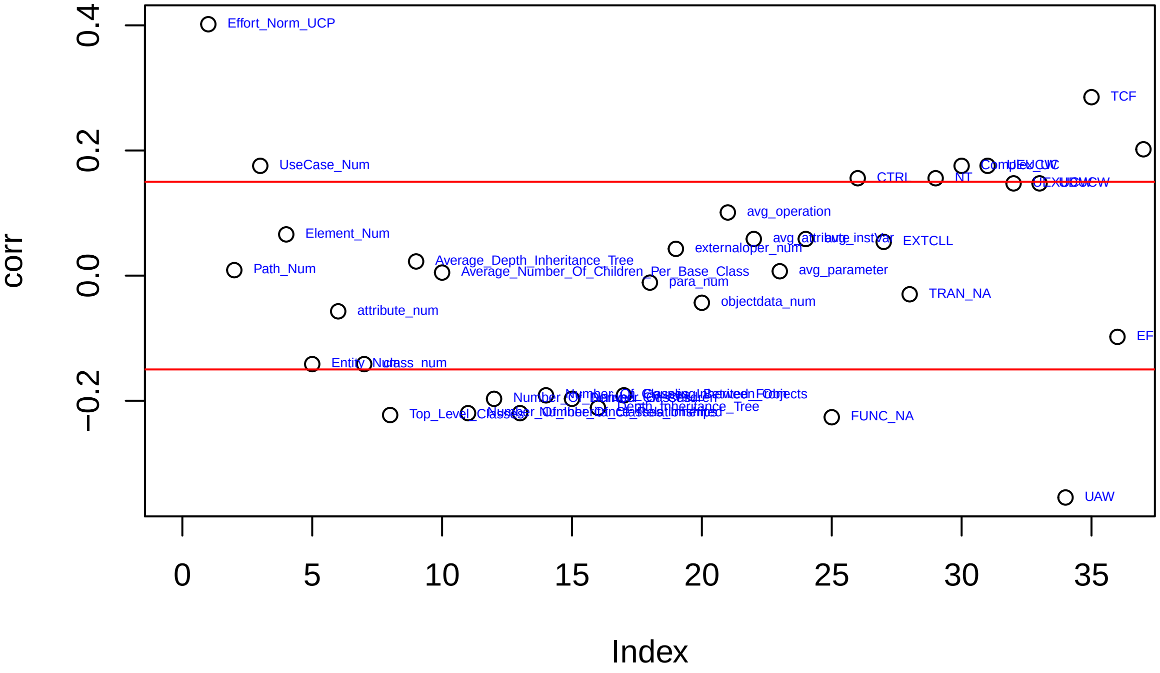
```
x <-data[,7:56];
x$Total_Degree<-NULL
x$operation_num<-NULL
x$usage_num<-NULL
x$real_num<-NULL
x$assoc_num<-NULL
x$EI<-NULL
x$INT<-NULL
x$DM<-NULL

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

```
##  Effort_Norm_UCP     Path_Num      UseCase_Num    Element_Num
##  Min.   : 140.6   Min.   : 27.0   Min.   : 5.0   Min.   : 59.0
##  1st Qu.: 507.0   1st Qu.: 55.0   1st Qu.: 8.0   1st Qu.:128.0
##  Median :1008.1   Median : 91.0   Median :11.0   Median :183.0
##  Mean   :1008.1   Mean   :129.0   Mean   :13.1   Mean   :209.6
##  3rd Qu.:1075.0   3rd Qu.:143.5   3rd Qu.:17.0   3rd Qu.:250.0
##  Max.   :3217.0   Max.   :488.0   Max.   :26.0   Max.   :555.0
##    Entity_Num    attribute_num      class_num     Top_Level_Classes
##  Min.   : 0.00   Min.   :  0.00   Min.   : 0.00   Min.   :  0.0
##  1st Qu.: 9.00   1st Qu.:  0.00   1st Qu.: 9.00   1st Qu.: 12.5
##  Median :11.00   Median : 13.00   Median :11.00   Median : 20.0
##  Mean   :15.42   Mean   : 39.26   Mean   :15.42   Mean   : 66.0
##  3rd Qu.:17.50   3rd Qu.: 57.50   3rd Qu.:17.50   3rd Qu.: 49.5
##  Max.   :49.00   Max.   :221.00   Max.   :49.00   Max.   :288.0
##  Average_Depth_Inheritance_Tree Average_Number_Of_Children_Per_Base_Class
##  Min.   :0.00000                Min.   :0.00000
##  1st Qu.:0.00000                1st Qu.:0.00000
##  Median :0.00000                Median :0.00000
##  Mean   :0.04285                Mean   :0.04236
##  3rd Qu.:0.09601                3rd Qu.:0.06452
##  Max.   :0.22414                Max.   :0.44000
##  Number_Of_Inheritance_Relationships Number_Of_Derived_Classes
##  Min.   : 0.000                      Min.   : 0.000
##  1st Qu.: 0.000                      1st Qu.: 0.000
##  Median : 0.000                      Median : 0.000
##  Mean   : 4.484                      Mean   : 3.774
##  3rd Qu.: 2.000                      3rd Qu.: 1.500
##  Max.   :24.000                      Max.   :22.000
##  Number_Of_Classes_Inherited Number_Of_Classes_Inherited_From
##  Min.   : 0.000              Min.   : 0.000
##  1st Qu.: 0.000              1st Qu.: 0.000
##  Median : 0.000              Median : 0.000
##  Mean   : 4.484              Mean   : 6.484
##  3rd Qu.: 2.000              3rd Qu.: 2.000
##  Max.   :24.000              Max.   :60.000
##  Number_Of_Children Depth_Inheritance_Tree Coupling_Between_Objects
##  Min.   : 0.000     Min.   : 0.000         Min.   : 0.000
##  1st Qu.: 0.000     1st Qu.: 0.000         1st Qu.: 0.000
##  Median : 0.000     Median : 0.000         Median : 0.000
##  Mean   : 3.774     Mean   : 5.258         Mean   : 6.484
##  3rd Qu.: 1.500     3rd Qu.: 2.000         3rd Qu.: 2.000
##  Max.   :22.000     Max.   :28.000         Max.   :60.000
##     para_num      externaloper_num objectdata_num   avg_operation
##  Min.   :  0.00   Min.   :  0.00   Min.   : 0.000   Min.   :0.0000
##  1st Qu.:  0.00   1st Qu.:  0.00   1st Qu.: 0.000   1st Qu.:0.0000
##  Median :  0.00   Median :  0.00   Median : 4.000   Median :0.0000
##  Mean   : 18.06   Mean   : 18.19   Mean   : 5.323   Mean   :0.6370
##  3rd Qu.:  3.00   3rd Qu.: 14.00   3rd Qu.: 6.000   3rd Qu.:0.8718
##  Max.   :197.00   Max.   :197.00   Max.   :33.000   Max.   :5.7941
##  avg_attribute    avg_parameter     avg_instVar        FUNC_NA
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   : 1.000
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.: 5.000
##  Median :0.9091   Median :0.0000   Median :0.9091   Median : 8.000
##  Mean   :1.6694   Mean   :0.5182   Mean   :1.6694   Mean   : 9.452
##  3rd Qu.:3.1553   3rd Qu.:0.1364   3rd Qu.:3.1553   3rd Qu.:10.500
##  Max.   :5.3333   Max.   :5.7941   Max.   :5.3333   Max.   :39.000
##       CTRL           EXTCLL         TRAN_NA           NT
##  Min.   :  1.00   Min.   : 0.00   Min.   :  0.0   Min.   :  1.00
##  1st Qu.: 14.00   1st Qu.: 9.00   1st Qu.: 35.0   1st Qu.: 14.00
##  Median : 29.00   Median :18.00   Median : 56.0   Median : 29.00
##  Mean   : 32.87   Mean   :22.84   Mean   : 96.1   Mean   : 32.87
##  3rd Qu.: 41.00   3rd Qu.:29.00   3rd Qu.:109.5   3rd Qu.: 41.00
##  Max.   :100.00   Max.   :79.00   Max.   :423.0   Max.   :100.00
##    Complex_UC       UEUCW           UEXUCW           UDUCW
##  Min.   : 5.0   Min.   : 75.0   Min.   :  2.00   Min.   :  2.00
##  1st Qu.: 8.0   1st Qu.:120.0   1st Qu.: 28.00   1st Qu.: 28.00
##  Median :11.0   Median :165.0   Median : 58.00   Median : 58.00
##  Mean   :13.1   Mean   :196.5   Mean   : 67.68   Mean   : 67.68
##  3rd Qu.:17.0   3rd Qu.:255.0   3rd Qu.: 86.50   3rd Qu.: 86.50
##  Max.   :26.0   Max.   :390.0   Max.   :227.00   Max.   :227.00
##       UAW              TCF               EF              EUCP
##  Min.   : 0.000   Min.   :0.7950   Min.   :0.8000   Min.   : 77.92
##  1st Qu.: 3.000   1st Qu.:0.8875   1st Qu.:0.9575   1st Qu.:127.55
##  Median : 6.000   Median :0.9250   Median :1.0000   Median :165.00
##  Mean   : 6.323   Mean   :0.9303   Mean   :0.9953   Mean   :185.46
##  3rd Qu.: 9.000   3rd Qu.:1.0000   3rd Qu.:1.0250   3rd Qu.:234.84
##  Max.   :12.000   Max.   :1.1350   Max.   :1.2500   Max.   :359.96
##      EXUCP             DUCP              SWTI          SWTII
##  Min.   :  9.64   Min.   :  9.64   Min.   : 270   Min.   : 290
##  1st Qu.: 31.21   1st Qu.: 31.21   1st Qu.: 550   1st Qu.: 825
##  Median : 56.93   Median : 56.93   Median : 910   Median :1244
##  Mean   : 66.57   Mean   : 66.57   Mean   :1290   Mean   :1851
##  3rd Qu.: 86.61   3rd Qu.: 86.61   3rd Qu.:1435   3rd Qu.:2145
##  Max.   :191.64   Max.   :191.64   Max.   :4880   Max.   :7234
##      SWTIII
##  Min.   : 214
##  1st Qu.: 650
##  Median : 982
##  Mean   :1473
##  3rd Qu.:1716
##  Max.   :5780
```

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(1:42,corr,row.names(corr),cex=0.4, pos=4, col="blue")
abline(h=0.15,col="red")
abline(h=-0.15,col="red")
```



Looking at the graph, following are the most correlated independent variables:
1. Effort_Norm_UCP 2. UseCase_Num 3. CTRL 4. NT 5. Complex_UC 6. UEUCW 7. TCF 8. EUCP 9. EXUCP 10. DUCP 11. Top_Level_Classes 12. Number_Of_Inheritance_Relationships 13. Number_Of_Derived_Classes 14. Number_Of_Classes_Inherited 15. Number_Of_Classes_Inherited_From 16. Number_Of_Children 17. Depth_Inheritance_Tree 18. Coupling_Between_Objects 19. FUNC_NA 20. UAW

Model Fitting

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

```
independentVar <- data.frame(x$Effort_Norm_UCP, x$UseCase_Num, x$CTRL, x$NT, x$Complex_UC, x$UEUCW, x$TC

names(independentVar)<- c("Effort_Norm_UCP","UseCase_Num","CTRL","NT","Complex_UC","UEUCW","TCF","EUCP"

#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:
##      Min       1Q   Median       3Q      Max
## -1759.21  -415.50    -0.05   444.83  1246.70
##
## Coefficients: (7 not defined because of singularities)
##                                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)                          -6292.7195  4421.6109  -1.423 0.172779
## Effort_Norm_UCP                          0.3669     0.2646   1.386 0.183525
## UseCase_Num                             41.2429   259.4861   0.159 0.875588
## CTRL                                    81.7000    64.7628   1.262 0.224155
## NT                                           NA         NA      NA       NA
## Complex_UC                                   NA         NA      NA       NA
## UEUCW                                        NA         NA      NA       NA
## TCF                                  10046.1378  5083.1322   1.976 0.064570
## EUCP                                    -5.8299    18.4307  -0.316 0.755616
## EXUCP                                  -43.6118    33.7428  -1.292 0.213484
## DUCP                                         NA         NA      NA       NA
## Top_Level_Classes                        2.7469    13.1427   0.209 0.836927
## Number_Of_Inheritance_Relationships -6531.0227  1302.5534  -5.014 0.000106
## Number_Of_Derived_Classes              616.5091   868.9024   0.710 0.487616
## Number_Of_Classes_Inherited                  NA         NA      NA       NA
## Number_Of_Classes_Inherited_From      -401.7018   265.8331  -1.511 0.149127
## Number_Of_Children                           NA         NA      NA       NA
## Depth_Inheritance_Tree                5538.6801  1349.9691   4.103 0.000742
## Coupling_Between_Objects                     NA         NA      NA       NA
## FUNC_NA                                -64.8707    30.6583  -2.116 0.049411
## UAW                                   -103.4690    58.7982  -1.760 0.096435
##
## (Intercept)
## Effort_Norm_UCP
## UseCase_Num
## CTRL
## NT
## Complex_UC
## UEUCW
## TCF                                 .
## EUCP
## EXUCP
## DUCP
## Top_Level_Classes
## Number_Of_Inheritance_Relationships ***
## Number_Of_Derived_Classes
## Number_Of_Classes_Inherited
## Number_Of_Classes_Inherited_From
## Number_Of_Children
## Depth_Inheritance_Tree              ***
## Coupling_Between_Objects
## FUNC_NA                             *
## UAW                                 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 834.5 on 17 degrees of freedom
## Multiple R-squared:  0.8214, Adjusted R-squared:  0.6848
## F-statistic: 6.013 on 13 and 17 DF,  p-value: 0.000429
```

```
raw_data <- read.csv(file = "/Users/mohit/Development/My Scripts/modelEvaluation_4_20_cleaned_1.csv", s
raw_data[is.na(raw_data[,"KSLOC"]), "KSLOC"] <- 0
```

```
X_data = subset(raw_data, select = c("KSLOC","Path_Num","UseCase_Num","Element_Num",
Y_data <- raw_data[,"Effort"]
```

```
library(glmnet)
```

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

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

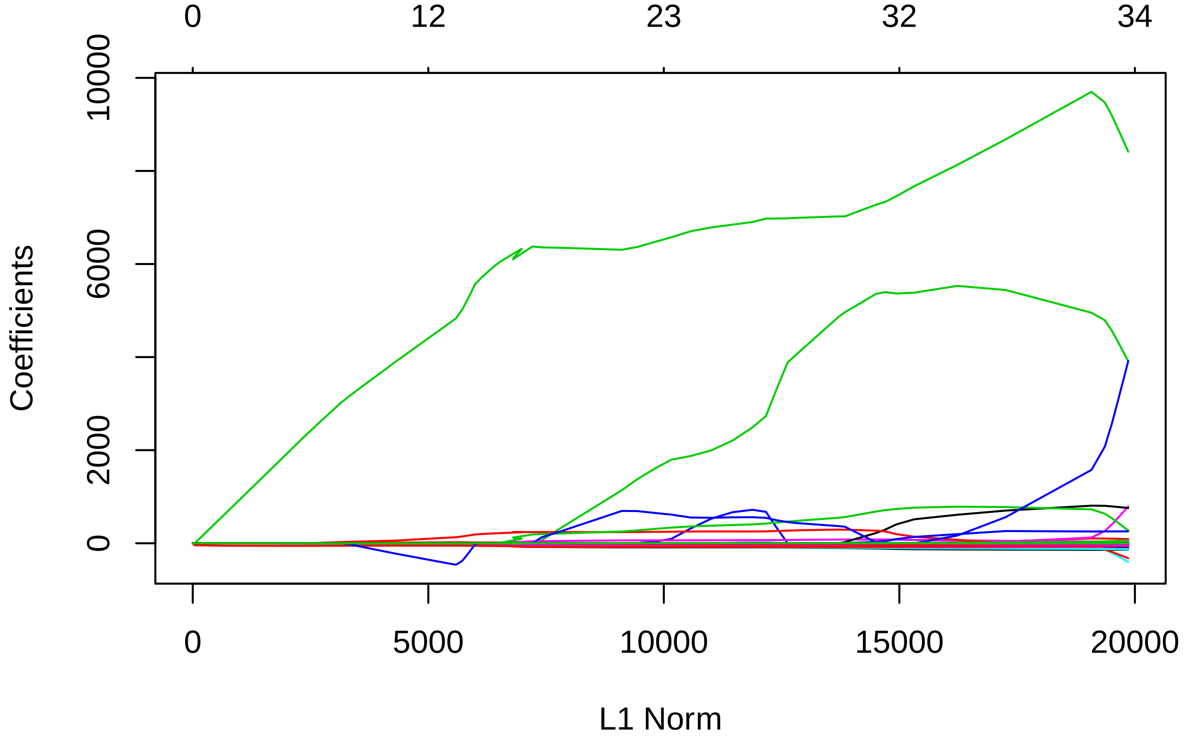
```
## Loaded glmnet 2.0-13
```

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

```
lasso_lm$lambda
```

```
##   [1] 518.491816 494.925563 472.430432 450.957740 430.461015 410.895897
##   [7] 392.220045 374.393038 357.376297 341.132993 325.627972 310.827678
##  [13] 296.700080 283.214603 270.342063 258.054599 246.325619 235.129739
##  [19] 224.442730 214.241461 204.503856 195.208840 186.336297 177.867025
##  [25] 169.782694 162.065809 154.699668 147.668329 140.956576 134.549882
##  [31] 128.434382 122.596841 117.024625 111.705676 106.628480 101.782051
##  [37]  97.155900  92.740014  88.524837  84.501247  80.660535  76.994389
##  [43]  73.494875  70.154420  66.965793  63.922094  61.016737  58.243432
##  [49]  55.596178  53.069246  50.657167  48.354720  46.156924  44.059020
##  [55]  42.056470  40.144938  38.320289  36.578573  34.916020  33.329033
##  [61]  31.814177  30.368174  28.987893  27.670349  26.412689  25.212191
##  [67]  24.066258  22.972410  21.928278  20.931604  19.980230  19.072098
##  [73]  18.205242  17.377785  16.587938  15.833991  15.114312  14.427343
##  [79]  13.771598  13.145657  12.548167  11.977833  11.433422  10.913756
##  [85]  10.417709   9.944208   9.492228   9.060792   8.648965   8.255856
##  [91]   7.880615   7.522428   7.180522   6.854157   6.542625   6.245252
##  [97]   5.961396   5.690441   5.431802   5.184918
```

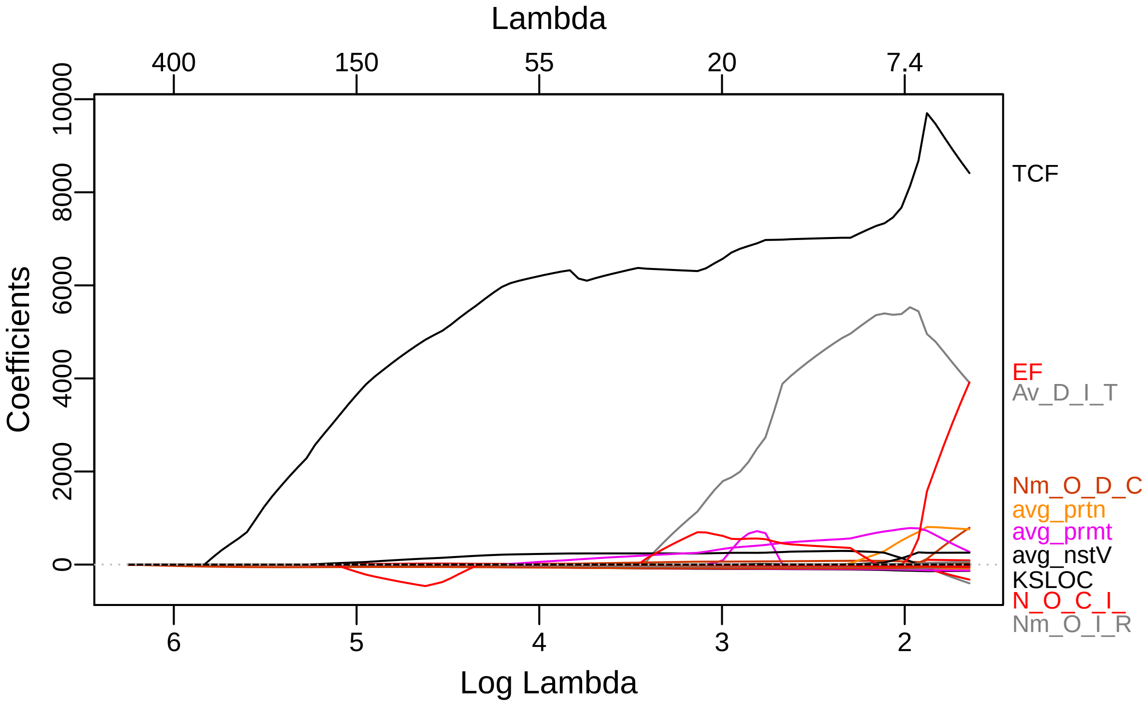
```
plot(lasso_lm)
```



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

```
## Warning: package 'plotmo' was built under R version 3.4.4

## Loading required package: plotrix

## Loading required package: TeachingDemos
```

```r
 #for 10 biggest final features
plot_glmnet(lasso_lm)                                   # default colors
```



```
#plot_glmnet(lasso_lm, label=10)
```

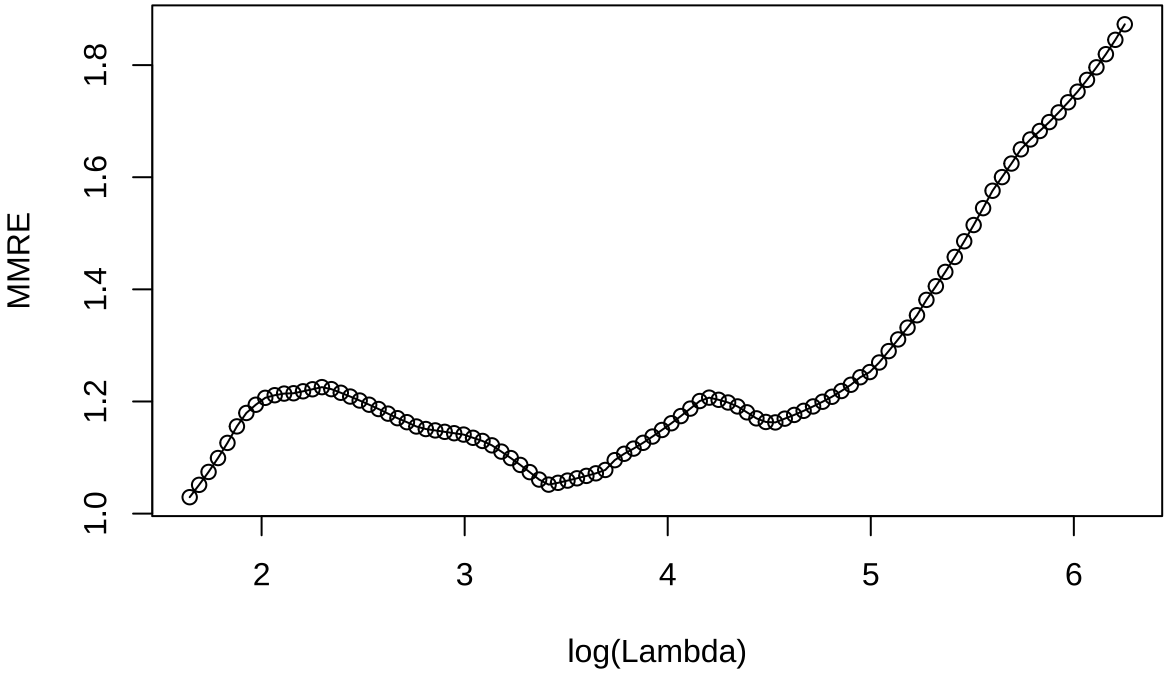
```
Lasso_range = function(x, y, k){
  # inputs:
      # x_matrix, a matrix containing independent variables
      # y: vector of dependent varaibles
      # k: the length of sequence
  # output:
      # seq: a sequence of lambdaa from high to low


  # define my own scale function to simulate that in glmnet
  myscale = function(x) sqrt(sum((x - mean(x)) ^ 2) / length(x))

  # normalize x
  sx = as.matrix(scale(x, scale = apply(x, 2, myscale)))
  # sy = as.vector(scale(y, scale = myscale(y)))
  max_lambda = max(abs(colSums(sx * as.vector(y)))) / dim(sx)[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(as.matrix(X_data),Y_data, 100)
```

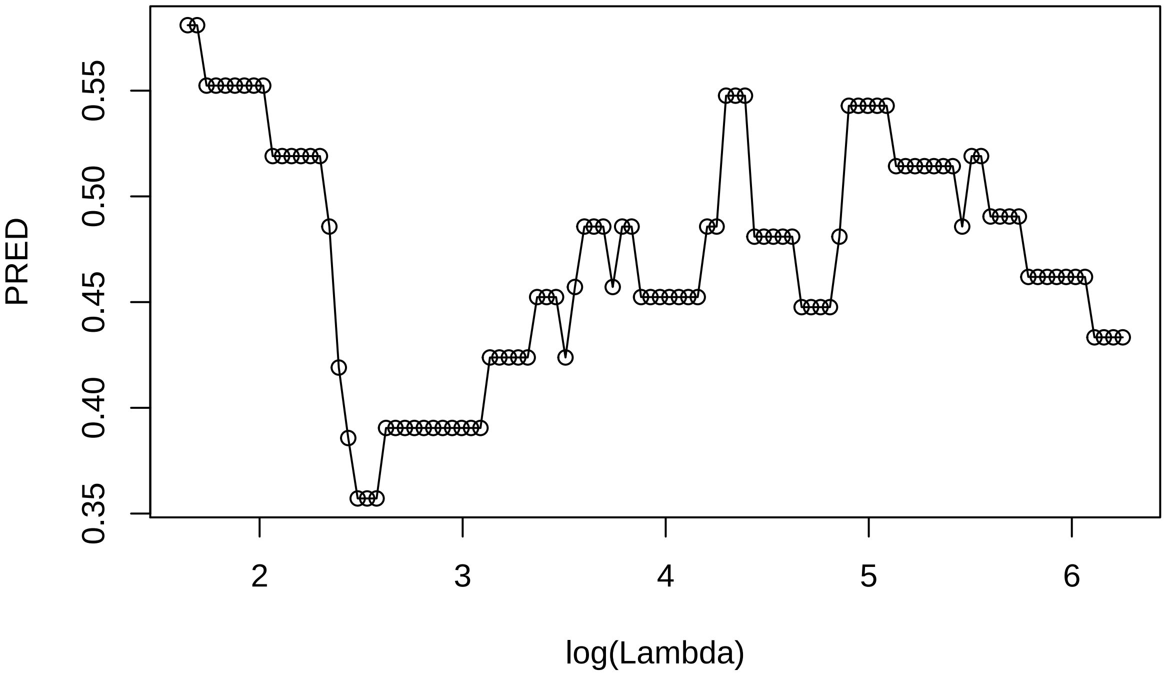
```
##   [1] 518.491816 494.925563 472.430432 450.957740 430.461015 410.895897
##   [7] 392.220045 374.393038 357.376297 341.132993 325.627972 310.827678
##  [13] 296.700080 283.214603 270.342063 258.054599 246.325619 235.129739
##  [19] 224.442730 214.241461 204.503856 195.208840 186.336297 177.867025
##  [25] 169.782694 162.065809 154.699668 147.668329 140.956576 134.549882
##  [31] 128.434382 122.596841 117.024625 111.705676 106.628480 101.782051
##  [37]  97.155900  92.740014  88.524837  84.501247  80.660535  76.994389
##  [43]  73.494875  70.154420  66.965793  63.922094  61.016737  58.243432
##  [49]  55.596178  53.069246  50.657167  48.354720  46.156924  44.059020
##  [55]  42.056470  40.144938  38.320289  36.578573  34.916020  33.329033
##  [61]  31.814177  30.368174  28.987893  27.670349  26.412689  25.212191
##  [67]  24.066258  22.972410  21.928278  20.931604  19.980230  19.072098
##  [73]  18.205242  17.377785  16.587938  15.833991  15.114312  14.427343
##  [79]  13.771598  13.145657  12.548167  11.977833  11.433422  10.913756
##  [85]  10.417709   9.944208   9.492228   9.060792   8.648965   8.255856
##  [91]   7.880615   7.522428   7.180522   6.854157   6.542625   6.245252
##  [97]   5.961396   5.690441   5.431802   5.184918
```

```
set.seed(2)
lambda_list <- Lasso_range(as.matrix(X_data),as.vector(Y_data),100)
percent = 50
cvfit = cv.glmnet(data.matrix(X_data),as.vector(Y_data),
                  standardize = T, 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(X_data)),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 <- Y_data[testIndexes]
     pred <- predict(cvfit,newx=as.matrix(X_data),s=lambda_list[[iterator]])
     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))
 }
```

```
plot(log(lambda_list),overall_mean_mmre,xlab="log(Lambda)",ylab="MMRE")
lines(log(lambda_list),overall_mean_mmre,xlim=range(log(lambda_list)), ylim=range(overall_mean_mmre), p
```

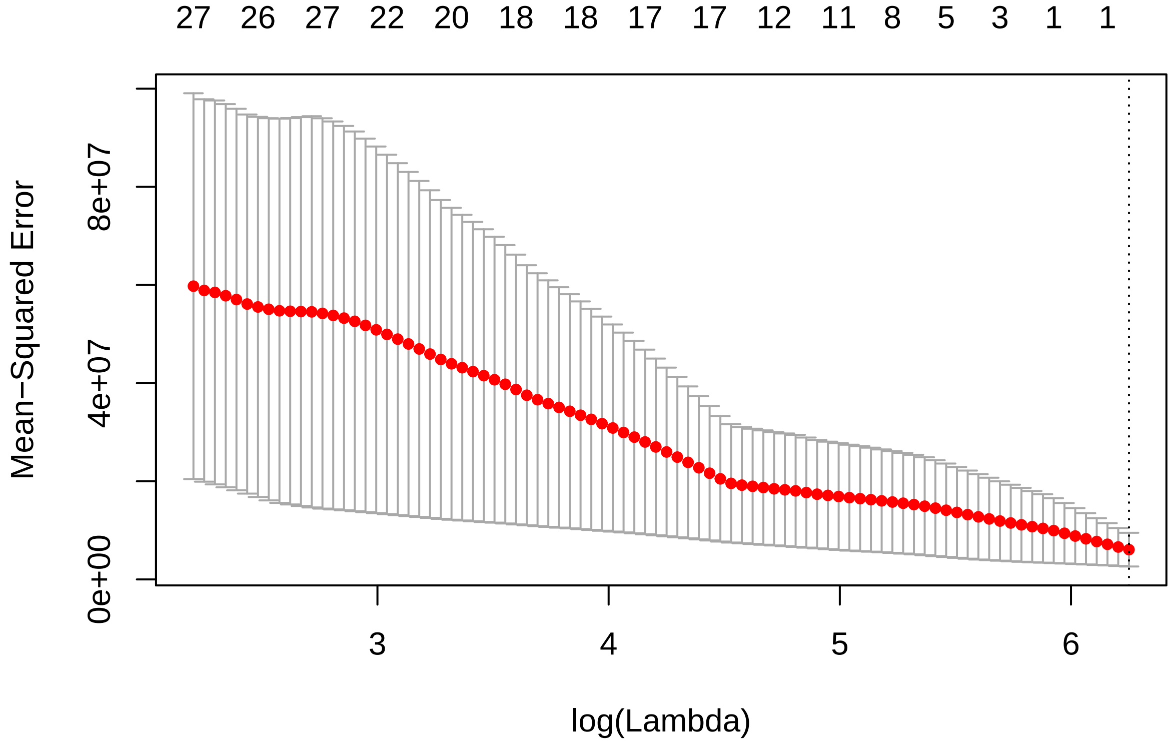


```
plot(log(lambda_list),overall_mean_pred,xlab="log(Lambda)",ylab = "PRED")
lines(log(lambda_list),overall_mean_pred,xlim=range(log(lambda_list)), ylim=range(overall_mean_pred), p
```



PRED
0.35
0.40
0.45
0.50
0.55
2
3
4
5
6
log(Lambda)

```
plot(cvfit)
```



27 26 27 22 20 18 18 17 17 12 11 8 5 3 1 1
Mean−Squared Error
0e+00
4e+07
8e+07
3
4
5
6
log(Lambda)