Feature Selection and Predictive Performance in Incomplete, Higher-Dimensional Datasets

Scenario 38: Simulate Datasets

Initial Setup

Load relevant packages and set a seed.

```
packs = c("caret","corrplot","dplyr","ggplot2","lmf","mvtnorm","reshape2")
lapply(packs, require, character.only = TRUE)
set.seed(103871)
```

Create Scenario Datasets

corr = mixed-high, p = 60, Complete Cases = 25%
Each scenario consists of 100 simulated datasets. 400 observations are simulated and split in half to produce training and testing datasets. For this scenario, the supervisor is continuous and is determined by: Y = $X\beta + \epsilon$, where X is a matrix of multivariate normal random variables with a mixed-high level of correlation (mean of .3 for most features with some features having a higher pairwise correlation between .75 and .85). The total number of features is 60 and consists of 10 truly associated features and 50 noise features. The first 10 features are the truly associated features. Features 1 through 5 are positively associated with the supervisor and have coefficients of .1, .2, .3, .4 and .5, respectively. Features 6 through 10 have the same coefficient magnitudes but are negatively associated with the supervisor. Complete datasets are first simulated and then missing data is incorporated (see below).

```r
scenID        = 38              # scenario no. for tracking and filenames
x             = 1:100           # no. of datasets to create for each scenario
n             = 400             # sample size per dataset before train/test split
p             = 60              # scenario dependent (35,60,110,210,260)
mu            = rep(0,p)
cor           = 0.3             # scenario dependent, initial level

Sigma         = matrix(rnorm(p*p, mean = 0.3, sd = 0.025), nrow = p, ncol = p)
diag(Sigma)   = 1

# Add highly correlated predictors
highCorrInd   = seq(3,p,15)
for (i in 1:length(highCorrInd)) {
  for (j in 0:2) {
   Sigma[highCorrInd[i]+j,highCorrInd[i]+j+1] = rnorm(1, mean = 0.85, sd = 0.02)}
  for (k in 0:1) {
   Sigma[highCorrInd[i]+k,highCorrInd[i]+k+2] = rnorm(1, mean = 0.75, sd = 0.02)}
  Sigma[highCorrInd[i],highCorrInd[i]+3] = rnorm(1, mean = 0.70, sd = 0.02)
}

Sigma[lower.tri(Sigma)] = t(Sigma)[lower.tri(Sigma)]
Sigma2        = nearPD(Sigma, corr = TRUE, keepDiag = TRUE)  # ensure matrix is pos. semidefinit
e

matFunc       = function(x) {rmvnorm(n,mu,Sigma2)}
compDataSet   = lapply(x,matFunc)
corrplot(cor(compDataSet[[1]]), tl.cex = 0.35)
```

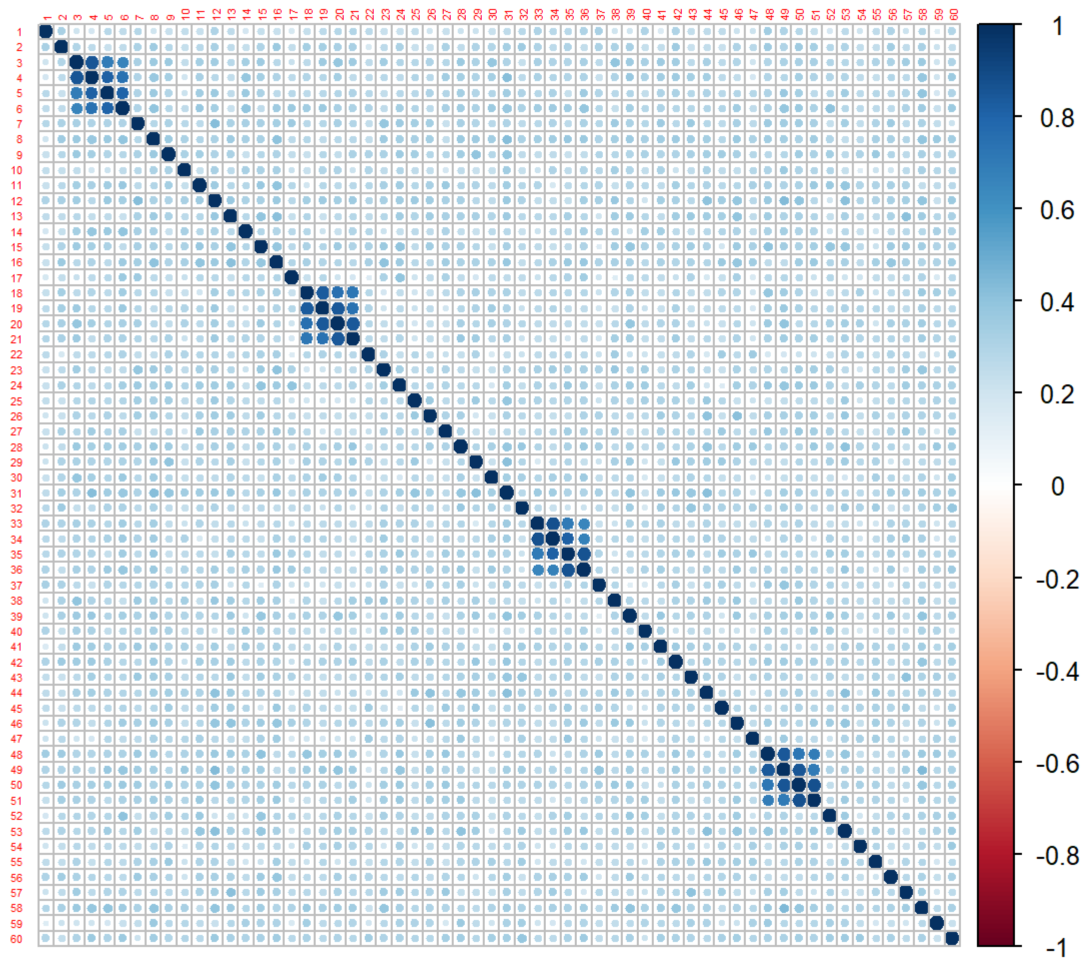

Fig. 1: Correlation plot for 1st sample dataset

Generate Supervisor Value Y and append to complete datasets

```
supVect             = c(.1,.2,.3,.4,.5,-.1,-.2,-.3,-.4,-.5,rep(0,p-10))
supFunc             = function(x) {x%*%supVect+rnorm(1)}

for (i in 1:length(x)){
   y = apply(compDataSet[[i]], 1, supFunc)
   compDataSet[[i]] = cbind(compDataSet[[i]],y)
}

names(compDataSet)  = paste0("CompMatrix", 1:length(x))
```

Generate Missing Indicators to create missing data

For this scenario, at least 75% of observations are set to contain at least one missing feature. Following Liu, et. al. (2016), Features #1 and 6 are kept as complete. Features #5 and 10 are set to missing at random where the missing probabilities are as follows:

$$prob_{missing}(x_5) = (1 + e^{-X_6+2.5})^{-1}$$

$$prob_{missing}(x_{10}) = (e^{-X_1-X_6+2})^{-1}$$

The remaining features are missing completely at random where the probability of being missing is set to result in datasets where at least 75% of all observations have at least one missing feature.

```r
missMatricesPrelim  = missMatricesFinal = vector("list", length(x))
for (i in 1:length(x)){
   missMatricesPrelim[[i]] = matrix(rbinom(n*(p-5+1), size = 1, .983), n, p-5+1)   # prob of non
missing, change by scenario
   missMatricesPrelim[[i]][missMatricesPrelim[[i]]==0] = NA
}

compVect = rep(1,n)

for (i in 1:length(x)){
  marVect1prob = 1-(1/(1+exp(-compDataSet[[i]][,1]-compDataSet[[i]][,6]+2)))  # MAR for predicto
r 5
  marVect1 = rbinom(n, size=1, prob = marVect1prob)
  marVect1[marVect1==0] = NA
  # Predictor 10 is MAR with components of x1 and x6
  marVect2prob = 1-(1/(1+exp(compDataSet[[i]][,1]+0.5*compDataSet[[i]][,6]+2)))
  marVect2 = rbinom(n, size=1, prob = marVect2prob)
  marVect2[marVect2==0] = NA
# combine genereated vectors
  missMatricesFinal[[i]] = cbind(compVect, missMatricesPrelim[[i]][,1:3],marVect1,compVect ,miss
MatricesPrelim[[i]][,4:6],marVect2,missMatricesPrelim[[i]][,7:(p-4)],compVect)
}
# add predictor labels
names(missMatricesFinal) = paste0("missMatrix", 1:length(x))
```

Create Missing Datasets

```r
missDataSet = vector("list", length(x))
for (i in 1:length(x)){
  missDataSet[[i]] = compDataSet[[i]]*missMatricesFinal[[i]]
}
```

Split into Training and Test Datasets

Use 50%/50% training test split.

```r
testMatrix = trainMatrix = matrix(0, nrow=length(x), ncol=200)
for (i in 1:length(x)){
  vect1            = sample(1:400, 200, replace = FALSE)
  trainMatrix[i,]   = vect1
  testMatrix[i,]  = (1:400)[-vect1]
}

trainDataSet = testDataSet = vector("list", length(x))
for (i in 1:length(x)){
  trainDataSet[[i]] = missDataSet[[i]][trainMatrix[1,],]
  testDataSet[[i]]  = missDataSet[[i]][testMatrix[1,],]
}
```

Check Percentage of Complete Observations (and Missing Percentage)

```r
ccFunc = function(x){as.numeric(summary(complete.cases(x))[3])}
nonMissingTrain = unlist(lapply(trainDataSet,ccFunc))
nonMissingTest = unlist(lapply(testDataSet,ccFunc))
```

Avg. % of Training Data Rows with Non Missing Values (complete cases): **23.8**
Avg. % of Test Data Rows with Non Missing Values (complete cases): **24.86**

Graphical Depiction of Missing Data - Dataset 1
A visual depiction of the missing features is shown below for one of the simulated training datasets. Lines in black indicate that the feature is missing for that observation. Features 5 and 10 contain a significant amount of missing data.

```r
ggplot_missing <- function(x){
  x %>% is.na %>% melt %>%
    ggplot(data = ., aes(x = Var2, y = Var1)) +
    geom_raster(aes(fill = value)) +
    scale_fill_manual(values = c("aliceblue","black"),labels = c("Present","Missing"))+
    theme_minimal() +
    theme(axis.text.x  = element_text(angle=90, vjust=0.25, size = 7)) +
    geom_vline(xintercept = 10.6, col = "blue") +
    labs(x = "Variables in Dataset", y = "Rows / observations")
}
ggplot_missing(as.data.frame(trainDataSet[[1]])[,1:p])
```

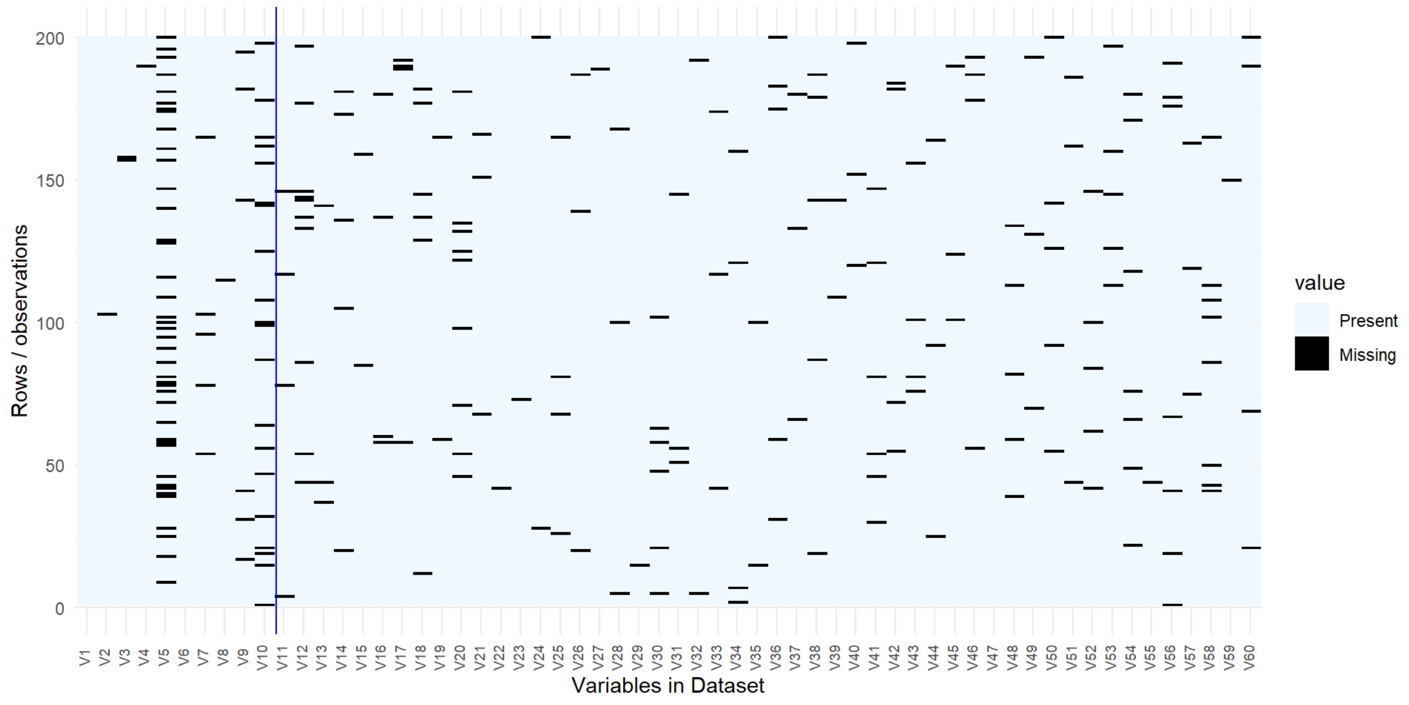

Fig. 2: Missing Data Plot for 1st Training Dataset

Save Datasets

Datasets are saved to be used by each of the 4 methods.

```r
save(compDataSet, file = paste0("compDataSet",scenID,".Rdata"))
save(missMatricesFinal, file = paste0("missMatricesFinal",scenID,".Rdata"))
save(missDataSet, file =paste0("missDataSet",scenID,".Rdata"))
save(trainDataSet, file = paste0("trainDataSet",scenID,".Rdata"))
save(testDataSet, file = paste0("testDataSet",scenID,".Rdata"))
```