"ISE 5103 Intelligent Data Analytics"
"Homework 7 - Modeling Competition"
"Daniel Carpenter, Sonaxy Mohanty, & Zachary Knepp"
"November 2022"

Packages --------

```
# Data Wrangling
library(tidyverse)
library(skimr)
library(lubridate) # dates

# Imputation
library(VIM)   # Factor: kNN
library(mice)  # Numeric: predictive mean matching

# Modeling
library(MASS)
library(caret) # Modeling variants like SVM
library(earth) # Modeling with Mars
library(adabag) # Modelling with AdaBoost
library(glmnet) # Modeling with LASSO
library(xgboost) #Modelling with Gradient boost

# Aesthetics
library(knitr)
library(cowplot)  # multiple ggplots on one plot with plot_grid()
library(scales)
library(kableExtra)
library(ggplot2)
library(inspectdf)

#Hold-out Validation
library(caTools)

#Data Correlation
library(GGally)
library(regclass)

#RMSE Calculation
library(MLmetrics)

#p-value for OLS model
library(broom)

#ncvTest
library(car)

# variable importance
library(vip)
#Partial plots
library(pdp)
```

 # Modeling

Building models

**** The below classifiers will be tested to classify the data:***

1. ***Logistic Regression***
2. ***LDA***
3. ***Classification and Regression Trees***
4. ***Elastic Net***
5. ***MARS***
6. ***Random Forest***
7. ***Boosted Trees - boosting, boosting.cv, gradient boosting***

Logistic Regression

**** This method was solely used to derive significant features for our model***

```
options("digits" = 6)
#resampling method
ctrl <- trainControl(method  = "cv",
                     number  = 10)
metric <- 'Accuracy'


#fit the model
# fit.logreg <- glm(readmitted~.,
#                   data=df.train.clean,
#                   family=binomial)

# step <- stepAIC(fit.logreg, direction="both", k=log(nrow(fit.logreg$data)))
# summary(step)


fit.logreg1 <- glm(readmitted ~ age + admission_source + time_in_hospital +
                     payer_code + num_lab_procedures + num_procedures + number_outpatient +
                     number_emergency + number_inpatient + diagnosis + number_diagnoses +
                     insulin + diabetesMed,
                   family = binomial,
                   data = df.train.clean)
```

LDA

```
# Fit the model

#pre-processing for LDA
preproc.param1 <- df.train.clean %>% preProcess(method = c("center", "scale"))
transformed1 <- preproc.param1 %>% predict(df.train.clean)

fit.lda2 <- train(readmitted ~ age + admission_source + time_in_hospital +
                    payer_code + num_lab_procedures + num_procedures + number_outpatient +
                    number_emergency + number_inpatient + diagnosis + number_diagnoses +
                    insulin + diabetesMed,
    data = transformed1,
    method="lda",
    metric=metric,
   # preProc    = c("center","scale"),
    trControl=ctrl,
    ) #0.6167489 0.2162356
```

```
# Key diagnostics
keyDiagnostics.lda <- data.frame(Model    = 'LDA',
                         Method    = 'lda',
                         Package = 'stats',
                         Hyperparameters = 'NA',
                         Selection = 'NA',
                         Accuracy = fit.lda2$results[,'Accuracy'],
                         Kappa = fit.lda2$results[,'Kappa'])


# Show output
keyDiagnostics.lda %>%
  knitr::kable()
```

Model	Method	Package	Hyperparameters	Selection	Accuracy	Kappa
LDA	lda	stats	NA	NA	0.616541	0.21582

 # CART

```
fit.cartf<- rpart(data=df.train.clean,
        readmitted ~ age + admission_source + time_in_hospital +
              payer_code + num_lab_procedures + num_procedures + number_outpatient +
              number_emergency + number_inpatient + diagnosis + number_diagnoses +
              insulin + diabetesMed,
        control=rpart.control(minsplit=10,cp=0.00073))

# pred.cart = predict(fit.cartf, type="prob")
# confusionMatrix(pred, df.train.clean$readmitted) #0.6283

# Key diagnostics
keyDiagnostics.cart <- data.frame(Model    = 'CART',
                         Method    = 'rpart',
                         Package = 'rpart',
                         Hyperparameters = 'cp',
                         Selection = 0.0007,
                         Accuracy = 0.6283,
                         Kappa = 0.2462)


# Show output
keyDiagnostics.cart %>%
  knitr::kable()

rm(acc, f1, p)
```

Model	Method	Package	Hyperparameters	Selection	Accuracy	Kappa
CART	rpart	rpart	cp	0.0007	0.6283	0.2462

 # Elastic Net

```
fit.elasticnet <- train(data = df.train.clean,
                readmitted~.,
                method     = "glmnet",          # Elastic net
                tuneLength = 10,                # 10 values of alpha and lambdas
                metric=metric,
                trControl  = ctrl) #0.6202403 0.22276121

get_best_result = function(caret_fit) {
  best = which(rownames(caret_fit$results) == rownames(caret_fit$bestTune))
  best_result = caret_fit$results[best, ]
  rownames(best_result) = NULL
  best_result
}
result.elasticnet <- get_best_result(fit.elasticnet)

hyperparameters.elasticnet = list('Alpha'  = result.elasticnet$alpha,
                          'Lambda' = result.elasticnet$lambda)


keyDiagnostics.elasticnet <- data.frame(Model    = 'Elastic Net',
                             Method    = 'glmnet',
                             Package = 'caret',
                             Hyperparameters = 'Alpha, Lambda',
                             Selection = paste('Alpha =',
                                               hyperparameters.elasticnet$Alpha, ',',
                                               'Lambda =',
                                               hyperparameters.elasticnet$Lambda),
                             Accuracy     = result.elasticnet$Accuracy,
                             Kappa = result.elasticnet$Kappa
                             )

# Show output
keyDiagnostics.elasticnet %>% knitr::kable()
```

Model	Method	Package	Hyperparameters	Selection	Accuracy	Kappa
Elastic Net	glmnet	caret	Alpha, Lambda	Alpha = 0.9 , Lambda = 0.00147817114982061	0.620586	0.224854

 # MARS

```
fit.mars <- train(data = df.train.clean,
             readmitted~.,
              method     = "earth",           # Earth is for MARS models
              tuneLength = 9,                 # 9 values of the cost function
              preProc    = c("center","scale"), # Center and scale data
              trControl  = ctrl
             ) #0.6257369 0.2375251


#hyperparameters
hyperparameters.mars = list('degree' = fit.mars[["bestTune"]][["degree"]],
                      'nprune' = fit.mars[["bestTune"]][["nprune"]])
# Key diagnostics
keyDiagnostics.mars <- data.frame(Model    = 'MARS',
                          Method    = 'earth',
                          Package = 'caret',
                          Hyperparameters = 'nprune, degree',
                          Selection = paste('Degree =', hyperparameters.mars$degree, ',',
                                            'nprune =', hyperparameters.mars$nprune),
                          Accuracy = fit.mars$results[9,'Accuracy'],
                          Kappa = fit.mars$results[9,'Kappa'])


# Show output
keyDiagnostics.mars %>%
  knitr::kable()
```

Model	Method	Package	Hyperparameters	Selection	Accuracy	Kappa
MARS	earth	caret	nprune, degree	Degree = 1 , nprune = 14	0.625944	0.237992

 # 3 Performance Evaluations of MARS model

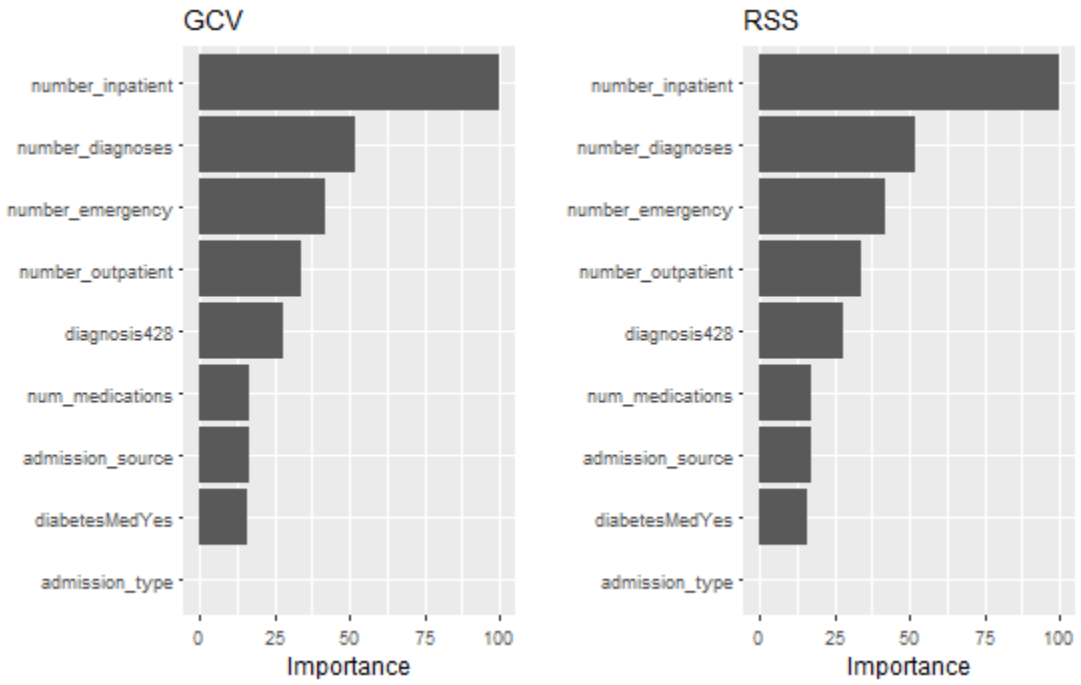
```
pred = predict(fit.mars, df.train.clean, type='raw') #type='class' if the model doesn't take raw
accuracy <- Accuracy(pred, df.train.clean$readmitted)
f1 <- F1_Score(pred, df.train.clean$readmitted)
precision <- Precision(pred, df.train.clean$readmitted)

cat("Accuracy: ", accuracy)
cat("\nF1 score: ", f1)
cat("\nPrecision: ", precision)
```

```
Accuracy:  0.626964
F1 score:  0.685366
Precision:  0.767367
```

Insight 1 of the MARS model - Best Variables

```
# variable importance plots
p1 <- vip(fit.mars, num_features = 40, bar = FALSE, value = "gcv") + ggtitle("GCV")
p2 <- vip(fit.mars, num_features = 40, bar = FALSE, value = "rss") + ggtitle("RSS")
gridExtra::grid.arrange(p1, p2, ncol = 2)
```



According to the MARS model, the best 3 predictor variables are number_inpatient, number_diagnosis, and number_emergency.
The importance of the predictors is measured in GCV (Generalized Cross-Validation), and RSS (Residual Sum of Squares)

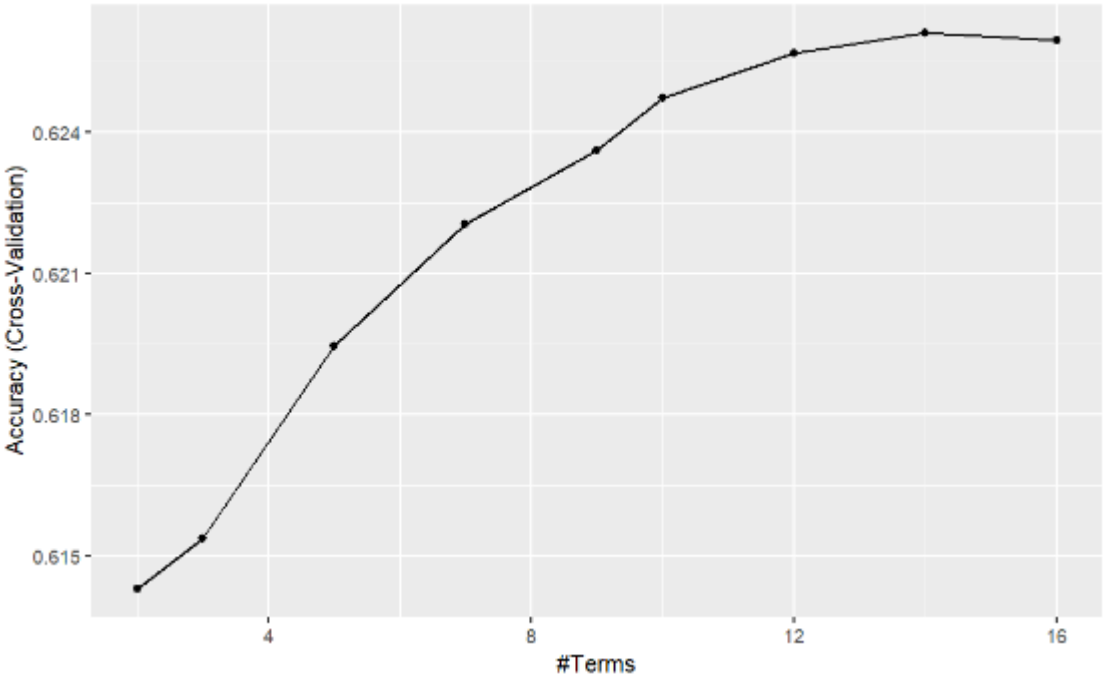
Insight 2 of the MARS model - Summary and Accuracy

summary(fit.mars)
ggplot(fit.mars)

```
GLM coefficients
                                 1
(Intercept)                1.225980
diagnosis428               0.107170
diabetesMedYes             0.081786
h(admission_type-1.34243)  0.364335
h(admission_type-2.71156) -1.019896
h(admission_source-0.0519679)  3.450142
h(0.29526-admission_source)    0.432903
h(admission_source-0.29526)   -3.619662
h(0.249163-num_medications)   -0.146009
h(0.465691-number_outpatient) -0.456027
h(1.87121-number_emergency)   -0.309985
h(1.098-number_inpatient)     -0.627504
h(number_inpatient-1.098)      0.232305
h(2.87903-number_diagnoses)   -0.133151

GLM (family binomial, link logit):
 nulldev    df       dev    df   devratio     AIC iters converged
   80003 57854     74629 57841      0.067   74700     4         1

Earth selected 14 of 19 terms, and 9 of 85 predictors (nprune=14)
Termination condition: RSq changed by less than 0.001 at 19 terms
Importance: number_inpatient, number_diagnoses, number_emergency, number_outpatient,
diagnosis428, admission_source, ...
Number of terms at each degree of interaction: 1 13 (additive model)
Earth GCV 0.227366    RSS 13142    GRSq 0.0873972    RSq 0.0882173
```

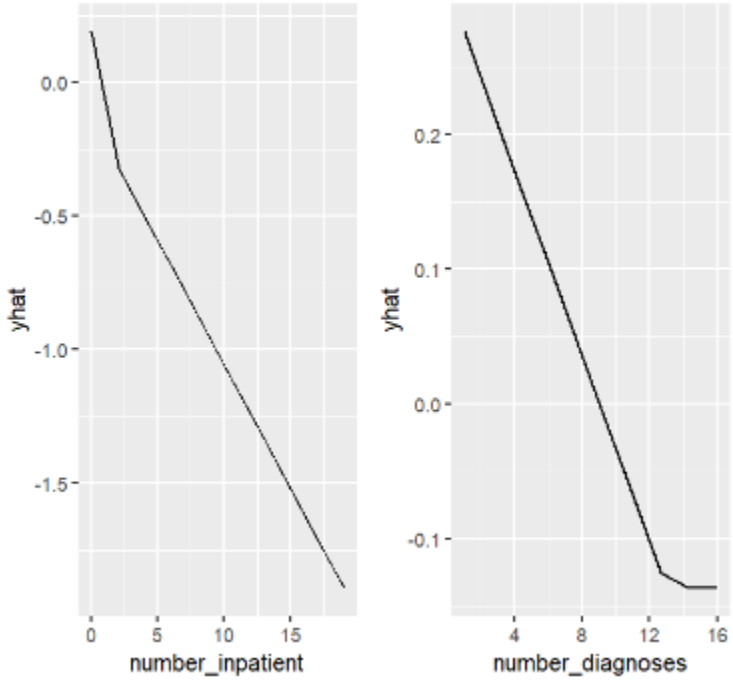


The summary of the MARS model shows the coefficients of the variables, and the plot shows the accuracy of the model as the number of predictors are increased. According to the summary table, the model picked 14 variables best represent the data.

Insight 3 of the MARS model - Partial Plots

```
p1 <- partial(fit.mars, pred.var = "number_inpatient", grid.resolution = 10) %>% autoplot()
p2 <- partial(fit.mars, pred.var = "number_diagnoses", grid.resolution = 10) %>% autoplot()
p3 <- partial(fit.mars, pred.var = c("number_inpatient", "number_diagnoses"), grid.resolution = 10) %>%
  plotPartial(levelplot = FALSE, zlab = "yhat", drape = TRUE, colorkey = TRUE, screen = list(z = -20, x = -60))

gridExtra::grid.arrange(p1, p2, p3, ncol = 3)
```



Partial Plots of the variables number_inpatient and number_diagnoses show the bends/spline that the mars model computes for the best fit. For example, 0-2 in the number_impatient graph represents 1 rate, while 2+ represents a different rate.

Random Forest

```
control <- trainControl(method = "cv",
   number = 10,
   search = "grid")

fit.rf <- train(readmitted ~ age + admission_source + time_in_hospital +
                payer_code + num_lab_procedures + num_procedures + number_outpatient +
                number_emergency + number_inpatient + diagnosis + number_diagnoses +
                insulin + diabetesMed,
            data=df.train.clean,
            method="rf",
            #tuneLength = 6,                # 9 values of the cost function
            #preProc    = c("center","scale"),
            metric=metric,
            trControl=control,
            allowParallel = TRUE) #0.6266357

#key diagnostics
keyDiagnostics.rf <- data.frame(Model    = 'Random Forest',
                        Method    = 'rf',
                        Package = 'caret',
                        Hyperparameters = 'mtry',
                        Selection = fit.rf$bestTune[,'mtry'],
                        Accuracy = fit.rf$results[1,'Accuracy'],
                        Kappa = fit.mars$results[1,'Kappa'])


# Show output
keyDiagnostics.rf %>%
  knitr::kable()
```

Model	Method	Package	Hyperparameters	Selection	Accuracy	Kappa
Random Forest	rf	caret	mtry	2	0.627379	0.21297

BOOSTING

```
fit.boost<-boosting(readmitted~ age + admission_source + time_in_hospital +
                payer_code + num_lab_procedures + num_procedures + number_outpatient +
                number_emergency + number_inpatient + diagnosis + number_diagnoses +
                insulin + diabetesMed,
               data = df.train.clean, boos = F, mfinal = 150) # 10 --> 0.6241
#50 --> 0.6253


#pred = predict(fit.boost, df.train.clean, type='raw')
# pred.btrain = predict(fit.boost, df.train.clean, type='prob')
# pred.btrain$error
#print(1-pred.btrain$error)
#accuracy and kappa calculation from conf matrix
#confusionMatrix(table(df.train.clean$readmitted, fit.boost$class))

# Key diagnostics
keyDiagnostics.boost <- data.frame(Model    = 'Boosting',
                        Method    = 'boosting',
                        Package = 'adabag',
                        Hyperparameters = 'mfinal',
                        Selection = 150,
                        Accuracy = 0.6253,
                        Kappa = 0.239)
# Show output
keyDiagnostics.boost %>%
  knitr::kable()

# boosting.cv
fit.cvmodel = boosting.cv(readmitted~age + admission_source + time_in_hospital +
                payer_code + num_lab_procedures + num_procedures + number_outpatient +
                number_emergency + number_inpatient + diagnosis + number_diagnoses +
                insulin + diabetesMed,
                data=df.train.clean,
                boos=FALSE,
                mfinal=50,
                v=5) #10 --> 0.6228848, BOOS-=TRUE #50 -->0.6254429, BOOS=FALSE


# print(1-fit.cvmodel[-1]$error)
# fit.cvmodel$error
# confusionMatrix(table(df.train.clean$readmitted, fit.cvmodel$class))

# Key diagnostics
keyDiagnostics.cvboost <- data.frame(Model    = 'Boosting.CV',
                        Method    = 'boosting.cv',
                        Package = 'adabag',
                        Hyperparameters = 'mfinal',
                        Selection = 50,
                        Accuracy = 0.6255,
                        Kappa = 0.2389)




keyDiagnostics.cvboost %>%
  knitr::kable()

## gradient boosting
fit.grboost <- train(readmitted~age + admission_source + time_in_hospital +
                payer_code + num_lab_procedures + num_procedures + number_outpatient +
                number_emergency + number_inpatient + diagnosis + number_diagnoses +
                insulin + diabetesMed,
                data=df.train.clean,
```

```
            method = "xgbTree",
            trControl = ctrl
 ) #0.6331347


#hyperparameters
hyperparameters.grboost = list('max_depth' = fit.grboost[["bestTune"]][["max_depth"]],
                    'eta' = fit.grboost[["bestTune"]][["eta"]],
                    'nrounds' = fit.grboost[["bestTune"]][["nrounds"]])
# Key diagnostics
keyDiagnostics.grboost <- data.frame(Model    = 'Gradient boost',
                        Method    = 'xgbTree',
                        Package = 'xgboost',
                        Hyperparameters = 'max_depth, eta, nrounds',
                        Selection = paste('max_depth =', hyperparameters.grboost$max_depth, ',',
                                          'eta =', hyperparameters.grboost$eta, ',',
                                      'nrounds=', hyperparameters.grboost$nrounds),
                        Accuracy = 0.6331347,
                        Kappa = 0.2563347)


# Show output
keyDiagnostics.grboost %>%
  knitr::kable()
```

Model	Method	Package	Hyperparameters	Selection	Accuracy	Kappa
Gradient boost	xgbTree	xgboost	max_depth, eta, nrounds	max_depth = 3 , eta = 0.3 , nrounds= 100	0.633135	0.256335

 # SUMMARY TABLE

```
# Add the key diagnostics here
rbind(
  #keyDiagnostics.logreg,
  keyDiagnostics.lda,
  keyDiagnostics.cart,
  keyDiagnostics.elasticnet,
  keyDiagnostics.mars,
  keyDiagnostics.rf,
  keyDiagnostics.boost,
  keyDiagnostics.cvboost,
  keyDiagnostics.grboost
) %>%

  # Round to 4 digits across numeric data
  mutate_if(is.numeric, round, digits = 4) %>%

  # Spit out kable table
  kable()
```

Model	Method	Package	Hyperparameters	Selection	Accuracy	Kappa
LDA	lda	stats	NA	NA	0.6165	0.2158
CART	rpart	rpart	cp	0.0007	0.6283	0.2462
Elastic Net	glmnet	caret	Alpha, Lambda	Alpha = 0.9 , Lambda = 0.00147817114982061	0.6206	0.2249
MARS	earth	caret	nprune, degree	Degree = 1 , nprune = 14	0.6259	0.2380
Random Forest	rf	caret	mtry	2	0.6274	0.2130
Boosting	boosting	adabag	mfinal	150	0.6253	0.2390
Boosting.CV	boosting.cv	adabag	mfinal	50	0.6255	0.2389
Gradient boost	xgbTree	xgboost	max_depth, eta, nrounds	max_depth = 3 , eta = 0.3 , nrounds= 100	0.6331	0.2563