**Machine Learning 2 Finals**

**Vivek D. Kulkarni 16th December 2017**

**1.** Supposedly, if you understand your discipline, you should be able to explain anything you do to your grandmother and she would understand. Explain what Machine Learning is, and how it differs from traditional statistics, in less than 100 words, not for your grandmother, but for a reasonably educated business professional who may be thinking of hiring you.

**Answer 🡪**

ML is backed by artificial intelligence and CS, on the other hand statistics has mathematical functions that formulizes relationship between two things. In ML we learn from data and try to predict / classify as per business problem. These algorithms with the help of programming and traditional statistics provides better solution without any human bias in decision making. ML banks on computing power and statistics on assumptions related to data and how distributed the data is. ML has open approach to learn and solve, however statistics has old-fashioned approach wherein it uses complex mathematical functions.

**2. Work with the diabetes data from women, PD.csv.**

npreg: the number of pregnancies

glucose: the plasma glucose concentration in an oral glucose tolerance test

bp: the diastolic blood pressure (mm Hg)

skinth: triceps skin-fold thickness measured in mm

bmi: the body mass index (weight in kg/(height in m)^2)

pedprob: the diabetes pedigree function, based on family history. Higher values indicate more relatives have diabetes.

age: age in years

Diabetes: Person has diabetes 5 years later, Yes or No

We want to come up with the best model to predict diabetes in a new sample.

a. Run all applicable ML methods using all variables. Be sure to run at a minimum Linear Regression, Linear Discriminant Analysis, Logistic regression, kNN, SVM, Naïve Bayes, Random forest, CART (Boosted/Bagged Decision Trees), Neural Network, Xgboost. Compile your results in a table. Display confusion matrix for each algorithm

b. Which ML method, of every applicable method, predicts the occurrence of diabetes best?

c. Which predictors perform best?

d. Based on your results, what recommendations would you make to the women to help them avoid diabetes?

Code:

diab <- read.csv("PD.csv")

diab1 <- diab

library(mice)

md.pattern(diab)

# bmi bp skinth

library(Hmisc)

diab$bp <- as.numeric(diab$bp)

diab$skinth <- as.numeric(diab$skinth)

diab$bmi <- as.numeric(diab$bmi)

diab$skinth <- impute(diab$skinth, mean)

diab$bmi <- impute(diab$bmi, mean)

diab$bp <- impute(diab$bp, mean)

# # a <-data.frame(apply(diab,2,function(x) { boxplot.stats(diab[x])$out}))

# "npreg" "glucose" "bp" "skinth" "bmi" "pedprob" "age"

# [8] "Diabetes"

#

diab$bp <- as.numeric(diab$bp)

diab$skinth <- as.numeric(diab$skinth)

diab$bmi <- as.numeric(diab$bmi)

########LDA############

library(caret)

require(MASS)

ED.R1 <- lda(Diabetes ~ .,data=diab)

pred <- predict(ED.R1,diab,type = 'class')

cm <- confusionMatrix(pred$class, diab$Diabetes)

Accuracies <- cm$overall["Accuracy"]

Accuracies

summary(Accuracies)

|  |
| --- |
| > Accuracies  Accuracy  0.7737342  > summary(Accuracies)  Min. 1st Qu. Median Mean 3rd Qu. Max.  0.7737 0.7737 0.7737 0.7737 0.7737 0.7737 |
|  |
| |  | | --- | | > | |

################kNN##############

model\_knn <- train(Diabetes ~ . ,data=diab, method='knn')

cm <- confusionMatrix(predict(model\_knn,type="raw",newdata = diab), diab$Diabetes)

Accuracies <- cm$overall["Accuracy"]

|  |
| --- |
| > Accuracies  Accuracy  0.7990506  > summary(Accuracies)  Min. 1st Qu. Median Mean 3rd Qu. Max.  0.7991 0.7991 0.7991 0.7991 0.7991 0.7991 |
|  |
| |  | | --- | | ################SVM ##############  library("e1071")  x <- svm(Diabetes ~ ., data=diab)  cm <- confusionMatrix(predict(x,diab), diab$Diabetes)  Accuracies <- cm$overall["Accuracy"]  > Accuracies  Accuracy  0.7990506  > summary(Accuracies)  Min. 1st Qu. Median Mean 3rd Qu. Max.  0.7991 0.7991 0.7991 0.7991 0.7991 0.7991 |   ################GBM ##############  library("e1071")  boostFit = train (Diabetes ~ .,method = "gbm", data = diab, verbose = FALSE)  cm <- confusionMatrix(predict(boostFit,diab), diab$Diabetes)  Accuracies <- cm$overall["Accuracy"]  > Accuracies  Accuracy  0.7911392  > summary(Accuracies)  Min. 1st Qu. Median Mean 3rd Qu. Max.  0.7911 0.7911 0.7911 0.7911 0.7911 0.7911  ###############ANN################  library(nnet)  modelfit <- multinom(Diabetes ~ ., data = diab, maxit=500, trace=T)  cm <- confusionMatrix(predict(modelfit,diab), diab$Diabetes)  Accuracies <- cm$overall["Accuracy"]  > Accuracies  Accuracy  0.7721519  > summary(Accuracies)  Min. 1st Qu. Median Mean 3rd Qu. Max.  0.7722 0.7722 0.7722 0.7722 0.7722 0.7722  ###############naiveBayes################  modelfit <- naiveBayes(Diabetes ~ ., data = diab,laplace=1)  cm <- confusionMatrix(predict(modelfit,diab) ,diab$Diabetes)  Accuracies <- cm$overall["Accuracy"]  Accuracy  0.7579114  > summary(Accuracies)  Min. 1st Qu. Median Mean 3rd Qu. Max.  0.7579 0.7579 0.7579 0.7579 0.7579 0.7579  ####################LOGISTIC##################################  aaa <- diab[,1:8]  modelfit <- glm(Diabetes ~ ., data = aaa,family = binomial(logit))  step(modelfit)  summ <-glm(Diabetes ~ npreg + glucose + bmi + pedprob,, data = aaa,family = binomial(logit))  modelfit <- glm(Diabetes ~ ., data = aaa,family = binomial(logit))  diab$pred <- predict(modelfit,type="response")  diab$Pre\_Diab <- ifelse(diab$pred>.5,"Yes","No")  cm <- confusionMatrix(diab$Pre\_Diab ,diab$Diabetes)  Accuracies <- cm$overall["Accuracy"]   |  | | --- | | > Accuracies  Accuracy  0.7721519  > summary(Accuracies)  Min. 1st Qu. Median Mean 3rd Qu. Max.  0.7722 0.7722 0.7722 0.7722 0.7722 0.7722 | |  | | **Results:**  **1) LDA – Accuracy**  0.7737342  **Confusion Matrix**  Reference Prediction No Yes  No 368 94  Yes 49 121  **2)KNN – Accuracy**  0.7990506  Reference Prediction No Yes  No 369 79  Yes 48 136  **3) SVM – Accuracy**  0.8132911  Reference Prediction No Yes  No 386 87  Yes 31 128  **#BEST MODEL for this data - {Best variables were**  **npreg, glucose, bmi, pedprob Performs better in model -- Suggestion- We should have controlled Glucose, Weight(BMI), Number of Pregnancy and Family Diabetes history.**  **}**  **4) GBM – Accuracy**  0.806962  Reference Prediction No Yes  No 383 88  Yes 34 127  **5) ANN – Accuracy**  0.7721519  Reference Prediction No Yes  No 368 95  Yes 49 120  **6) Naïve – Bayes – Accuracy**  0.7579114  Reference Prediction No Yes  No 342 78  Yes 75 137  **7) GLM (Log) – Accuracy**  0.7721519  Reference Prediction No Yes  No 368 95  Yes 49 120     |  | | --- | |  | | |  | |

**3. Work with the NIJ\_BMOP.csv data.**

We want to come up with the best model to predict "out" (the outcome) in a new sample.

a. Describe the predictors (p1 through p91). Screen them for too many missing values, which will limit the sample sizes. Ideally, you want as many predictors without sacrificing sample size or distribution of your outcome.

b. Run all applicable ML methods. Be sure to run at a minimum Linear Regression, Linear Discriminant Analysis, kNN, SVM, Random Forest, CART (Boosted/Bagged Decision Trees), Neural Network, Xgboost, Partial Least Square Regression, Principal Component Regression, and Generalized Additive Model. Remove redundant or noisy predictors as necessary. Compile your results in a table. Display confusion matrix for each algorithm.

c. Which ML method, of every applicable method, predicts the outcome best?

d. Which predictors help predict the best?

e. Which ML method was *easiest* to use on these data?

**Answer 🡪**

BMOP <- read.csv("NIJ\_BMOP.csv")

BMOP1 <- BMOP

library(mice)

md.pattern(BMOP[,2:10])

BMOP <- BMOP[,-5]

md.pattern(BMOP[,10:20])

BMOP <- BMOP[,-14]

md.pattern(BMOP[,20:30])

BMOP <- BMOP[,-23]

md.pattern(BMOP[,30:35])

#p35 p34 p36

BMOP <- BMOP[,- (33:35)]

md.pattern(BMOP[,35:40])

BMOP <- BMOP[,-39]

md.pattern(BMOP[,40:50])

BMOP <- BMOP[,-39]

md.pattern(BMOP[,50:60])

#p62 p55

BMOP <- BMOP[,-c(57,50)]

md.pattern(BMOP[,60:70])

#p75

BMOP <- BMOP[,-68]

md.pattern(BMOP1[,80:93])

drop <-c("p75","p80" ,"p82", "p83", "p81", "p79", "p86", "p90")

BMOP <- BMOP[,!names(BMOP) %in% drop]

a <- BMOP[,2:74]

a <- data.frame(apply(a,2,as.numeric))

a <- apply(a, 2, function(x){x <- replace(x, is.na(x), mean(x, na.rm=TRUE))})

BMOP2 <- data.frame(a)

###############GBM################

set.seed (32323)

library(caret)

trCtrl = trainControl (method = "cv", number = 10)

boostFit = train (out ~ ., trControl = trCtrl,

method = "gbm", data = BMOP2, verbose = FALSE)

boostFit

Stochastic Gradient Boosting

357 samples

72 predictor

No pre-processing

Resampling: Cross-Validated (10 fold)

Summary of sample sizes: 321, 322, 321, 321, 320, 321, ...

Resampling results across tuning parameters:

interaction.depth n.trees RMSE Rsquared MAE

1 50 9.068572 0.8445477 7.209373

1 100 8.679993 0.8527335 6.830598

1 150 8.671503 0.8523864 6.817886

2 50 8.669992 0.8515084 6.704931

2 100 8.611574 0.8532881 6.692013

2 150 8.532806 0.8553271 6.609574

3 50 8.384127 0.8614982 6.417377

3 100 8.417842 0.8595987 6.460769

3 150 8.556132 0.8545373 6.590594

Tuning parameter 'shrinkage' was held constant at a value of 0.1

Tuning parameter 'n.minobsinnode' was held constant at a value of 10

RMSE was used to select the optimal model using the smallest value.

The final values used for the model were n.trees = 50,

interaction.depth = 3, shrinkage = 0.1 and n.minobsinnode = 10.

###############LDA################

library(MASS)

ED.R1 <- lda(out ~ .,data=BMOP2)

cm <- confusionMatrix (BMOP2$out, predict (ED.R1, BMOP2)$class)

Accuracies <- cm$overall["Accuracy"]

Accuracies

###############LDA################

Accuracy

0.7703081

model\_knn <- train(out ~ .,data=BMOP2, method='knn')

model\_knn

cm <- confusionMatrix(predict(model\_knn,type="raw",newdata=BMOP2), diab$Diabetes)

Accuracies <- cm$overall["Accuracy"]

Accuracies

summary(Accuracies)

Error in confusionMatrix.default(predict(model\_knn, type = "raw", newdata = BMOP2), :

the data cannot have more levels than the reference

###############ANN################

library(nnet)

modelfit <- multinom(out ~ .,data=BMOP2,maxit=5773, MaxNWts=6000,trace=F)

cm <- confusionMatrix (BMOP2$out, predict (modelfit, BMOP2))

Accuracies <- cm$overall["Accuracy"]

Accuracies

###############naiveBayes################

library("e1071")

nb\_laplace1 <- naiveBayes(out ~ .,data=BMOP2, laplace=1)

cm <- confusionMatrix (BMOP2$out, predict (nb\_laplace1, BMOP2))

RMSE = sqrt(mean((BMOP2$out - Yhat)^2))

###############SVM################

svm\_model <- svm(out ~ .,data=BMOP2)

AIC(svm\_model)

svm\_model

###############SVM################

###################Xgboost########

LinearModl <- lm(out ~ .,data=BMOP2)

stepAIC(LinearModl)

library(xgboost)

dtrain <- xgb.DMatrix(data = BMOP2[,2:73],label = BMOP2[,1])

xgb <- xgboost(data = BMOP2[,2:73],

label = BMOP2[,1],

eta = 0.1,

max\_depth = 15,

nround=25,

subsample = 0.5,

colsample\_bytree = 0.5,

seed = 1,

num\_class = 12,

nthread = 3

)

# not working on this data

#######Partial Least Square Regression#########

library(plsdepot)

pls1 = plsreg1(BMOP2[,2:73], BMOP2[,1], comps = 3)

pls1$R2

plot(pls1)

##############Principal Component Regression###############

require(pls)

pcr\_model <- pcr(out ~ .,data=BMOP2, scale = TRUE, validation = "CV")

summary(pcr\_model)

##### Generalized Additive Model#########

require(gam)

gam1 <- gam(out ~ .,data=BMOP2)

summary(gam1)

**Results :**

**1) LDA – Accuracy**

0.7703

Kappa : 0.7666

**2) KNN – R2**

0.1940216

**RMSE -** 20.34051 for K = 9

3) **SVM**

gamma: 0.01388889

epsilon: 0.1

Number of Support Vectors: 298

4) **GBM**

**Accuracy -** 0.8642554

**RMSE -** 8.204637

NTrees – 50

**This is best model as per data. The best variables are p4 + p6 + p9 + p10 + p14 + p18 + p21 + p24 +   
 p25 + p26 + p27 + p28 + p31 + p37 + p39 + p40 + p46 + p47 +   
 p48 + p52 + p54 + p57 + p59 + p60 + p61 + p65 + p68 + p69 +   
 p74 + p78 + p85 + p89**

**5) ANN (Artificial Neural Network)**

AIC: 11242

Not fitting

**6) Naïve Bayes -** Not Working on this Set

**7) LM (Linear )**

AIC=1506.79

**8) XgBoost**

**Not working on this set**

**9) Partial Least Square Regression**

t1 t2 t3   
0.83518466 0.03523964 0.01040633

**10)** Principal Component Regression

Data: X dimension: 357 72   
 Y dimension: 357 1  
Fit method: svdpc

**11)** Generalized Additive Model

AIC: 2577.138

Number of Local Scoring Iterations: 2