Team name: Anonymous

Team members: Xinyue Xing, Boyu Liu, Yingxi Huang, Tianyi Li, Yingjie Gao

Date: April 6 2022

This journal has to be uploaded to 1) a dropbox folder (**link**) **AND** 2) posted to your project website every two weeks by 11:59 pm on the second Friday. The first due date is Mar. 4th, Mar. 25th and the third due date is April 7th. When uploading to the dropbox folder, change the file name to "Team_name_MM-DD-2022" where MM = month and DD= day of upload).

Team roles for this report (write down name):

Facilitator(s): Tianyi Li

Recorder(s): Yingxi Huang

Deliverer(s): Xinyue Xing

Planner(s): Yingjie Gao

Team contact: Boyu Liu

See last page for description of roles. Obviously one person can take more than one role or there can be more than one person per role or make your own roles!

1. **Describe briefly what the main goal of your team is (so the peer reviewer has some context). E.g. we are working on image classification for blah de blah. Our goal is blah de blah etc. In the initial part of the semester before your proposal it is ok to put down "we are still coming up with ideas on team project".**

After some initial discussion, we decided to experiment with four different classification algorithms in this meeting, including KNN, logistic, LDA, and QDA. Specifically, Yingjie Gao was responsible for the logistic model; Tianyi Li run the initial LDA; Xinyue Xing tried QDA; Boyu Liu conducted KNN. Our goal is to have an initial understanding and observations of the performance of the four models.

I. **What was done during the report period regarding the project:** If you want to include code include this in the Appendix. Describe what the group did (including contributions of individual team members) with regards to the group project during this report period. Give enough details so I understand what you folks have been doing over the week. Include dates of your meeting(s) and who met on these days.

Meeting 4 - 3/30/2022 - All members attended

- Conducted via zoom
- Discussed how to maintain symmetry in our prediction
- Cleaned data: changed game results data into binary format

Meeting 5 - 4/06/2022 - All members attended

- Met in person
- Discussed future plan
- Divided work among team members
- Started to build four classification models
- Started to work on presentation slides

II. **What were obstacles faced if any in working on the project?** This could be technical (like not being able to implement or understand particular techniques) or time issues (midterms for other courses etc).

We found that it is not easy to achieve consensus on time arrangements, especially last week. All members had several midterms, and we were not sure what was the best approach to follow up with the project. Fortunately, we maintained effective and continuous communication and arranged a zoom meeting. One key issue is that we are not familiar with the models. We understand the models theoretically, but we are not sure how to implement those models with R codes. Since many of the homework solutions are not posted yet, we cannot find many references to guide us through the coding process.

III. What is the plan for the next reporting period including what each team member is planning to work on. Describe goals and potential timelines (" I plan to finish understanding x to see if it can be implemented for our project by Wednesday etc".)

Our initial models yield high level of accuracy. QDA has an accuracy of 97% and a ROC of 99%. LDA has an accuracy of 88%. Logistic has an accuracy of 88%. KNN has an accuracy of 90%. We are trying to make sense of the high results by examining potential limitations. In addition, we are planning to adjust the current models and experiment other models. Moreover, we will continue to work on the presentation slides, with a focus on the background parts.

IV. Appendix

```
# LDA
# clean data for LDA
```{r}
# delete all the categorical variables
train_LDA = select(train, -c(1:6, 14:17, 25:26))

# fit LDA
library(MASS)
fit.lda <- lda(binaryResult ~ ., data = train_LDA)
fit.lda
```

test data
```{r}
# same processing for test data - delete categorical variables
test_LDA = dplyr::select(test, -c(1:6, 14:17, 25:26))

# Test
library(pROC)
fit.lda.fitted.test <- (predict(fit.lda, test_LDA, type="response")$posterior)[,2]
fit.lda.test.roc <- roc(test_LDA$binaryResult, fit.lda.fitted.test)
lda.fit.test = ifelse(fit.lda.fitted.test>0.5, 1, 0)
lda.tab = table(lda.fit.test,test_LDA$binaryResult)
lda.tab

# Accuracy
library("caret")
(lda.tab[1,1]+lda.tab[2,2])/sum(lda.tab)

confusionMatrix(data = as.factor(lda.fit.test),
                reference = as.factor(test_LDA$binaryResult))
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 4710  655
         1  511 4124

               Accuracy : 0.8834
                 95% CI : (0.8769, 0.8896)
    No Information Rate : 0.5221
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.766

 Mcnemar's Test P-Value : 2.817e-05

            Sensitivity : 0.9021
            Specificity : 0.8629
         Pos Pred Value : 0.8779
         Neg Pred Value : 0.8898
             Prevalence : 0.5221
         Detection Rate : 0.4710
   Detection Prevalence : 0.5365
      Balanced Accuracy : 0.8825

       'Positive' Class : 0
```
```

```r
library(MASS)
# remove variables that are binary
qda.train <- subset(train, select =c(-Legendary.x,-Legendary.y,-Result,-Name.x,-Name.y,-Type.1.x, -Type.1.y,-Type.2.x,-Type.2.y))
# remove variables that are collinear
tmp <- cor(qda.train)
tmp[upper.tri(tmp)] <- 0
diag(tmp) <- 0
qda.train <- qda.train[,!apply(tmp,2, function(x) any(abs(x) > 0.9, na.rm = TRUE))]
head(qda.train)
# remove variables in test dataset
qda.test <- subset(test, select = colnames(qda.train))
qda.mod <- qda(binaryResult~., data = qda.train)
summary(qda.mod)

# fit the model
qda.pred <- predict(qda.mod, newdata = test,type="response")

#QDA Accuracy, Sensitivity, and Specificity
confusionMatrix(data = as.factor(qda.pred$class),
                reference = as.factor(qda.test$binaryResult))

#QDA ROC and AUC
prediction(qda.pred$posterior[,2], qda.test$binaryResult) %>%
  performance(measure = "tpr", x.measure = "fpr") %>%
  plot()

prediction(qda.pred$posterior[,2], qda.test$binaryResult) %>%
  performance(measure = "auc") %>%
  .@y.values
```

```
          Reference
Prediction    0    1
         0 5099  169
         1  122 4610

               Accuracy : 0.9709
                 95% CI : (0.9674, 0.9741)
    No Information Rate : 0.5221
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9417

 Mcnemar's Test P-Value : 0.007006

            Sensitivity : 0.9766
            Specificity : 0.9646
         Pos Pred Value : 0.9679
         Neg Pred Value : 0.9742
             Prevalence : 0.5221
         Detection Rate : 0.5099
   Detection Prevalence : 0.5268
      Balanced Accuracy : 0.9706

       'Positive' Class : 0
```

```r
#KNN
library(class)
knn_models <- list()
ktrain <- subset(qda.train, select = -c(binaryResult))
ktest <- subset(qda.test, select = -c(binaryResult))
head(qda.train)

ktrain.labels <- qda.train[,16]
ktest.labels <- qda.test[,16]

confusion <- function(yhat, y, quietly = FALSE){
  confusion_mat <- table(yhat, y, deparse.level = 2)
  stats <- data.frame(sensitivity = confusion_mat[1, 1]/sum(confusion_mat[, 1]),
                      specificity = confusion_mat[2, 2]/sum(confusion_mat[, 2]))
  return(list(confusion_mat = confusion_mat, stats = stats))
}
knn <- knn(train = ktrain, test = ktest, cl = qda.train$binaryResult, k = 50)
#KNN Accuracy
mean(knn == qda.test$binaryResult)

#KNN Sensitivity and Specificity
confusion(yhat = knn, y = qda.test$binaryResult, quietly = T)
```

```
[1] 0.8978
$confusion_mat
    y
yhat    0    1
   0 4793  594
   1  428 4185
```

KNN

```
set.seed(123) #randomization`

 #creating indices
trainIndex <- createDataPartition(combatm$binaryResult,p=0.8,list=FALSE)

 #splitting data into training/testing data using the trainIndex object
pokemon_train <- combatm[trainIndex,] #training data (80% of data)

pokemon_test <- combatm[-trainIndex,] #testing data (20% of data)

pokemon_train_1 <- pokemon_train[, -which(names(pokemon_train) %in%
c("First_pokemon", "Second_pokemon", "Winner",  "Generation.x", "Name.x", "Type.1.x",
"Type.2.x", "Generation.y", "Name.y", "Type.1.y", "Type.2.y", "Legendary.x", "Legendary.y",
"Result"))]

pokemon_test_1 <- pokemon_test[, -which(names(pokemon_train) %in% c("First_pokemon",
"Second_pokemon", "Winner",  "Generation.x", "Name.x", "Type.1.x", "Type.2.x",
"Generation.y", "Name.y", "Type.1.y", "Type.2.y", "Legendary.x", "Legendary.y", "Result"))]

na.omit(combatm)
nor <-function(x) { (x -min(x))/(max(x)-min(x))}
pokenorm_train <- as.data.frame(lapply(pokemon_train_1, nor))
pokenorm_test <- as.data.frame(lapply(pokemon_test_1, nor))

library(class)
poketrain_labels <- pokenorm_train[,13]
poketest_labels <- pokenorm_test[,13]
knn.10 <- knn(train=pokenorm_train, test=pokenorm_test, cl=poketrain_labels,k=10)
knn.cm <- table(knn.10, poketest_labels)
accuracy <- function(x){sum(diag(x)/(sum(rowSums(x)))) * 100}
accuracy(knn.cm)
```

sensitivity <dbl>	specificity <dbl>
0.9180234	0.8757062

```r
library(MASS)
# remove variables that are binary
qda.train <- subset(train, select =c(-Legendary.x,-Legendary.y,-Result,-Name.x,-Name.y,-Type.1.x, -Type.1.y,-Type.2.x,-Type.2.y))
# remove variables that are collinear
tmp <- cor(qda.train)
tmp[upper.tri(tmp)] <- 0
diag(tmp) <- 0
qda.train <- qda.train[,!apply(tmp,2, function(x) any(abs(x) > 0.9, na.rm = TRUE))]
head(qda.train)
# remove variables in test dataset
qda.test <- subset(test, select = colnames(qda.train))
qda.mod <- qda(binaryResult~., data = qda.train)
summary(qda.mod)

# fit the model
qda.pred <- predict(qda.mod, newdata = test,type="response")

#QDA Accuracy, Sensitivity, and Specificity
confusionMatrix(data = as.factor(qda.pred$class),
                reference = as.factor(qda.test$binaryResult))

#QDA ROC and AUC
prediction(qda.pred$posterior[,2], qda.test$binaryResult) %>%
  performance(measure = "tpr", x.measure = "fpr") %>%
  plot()

prediction(qda.pred$posterior[,2], qda.test$binaryResult) %>%
  performance(measure = "auc") %>%
  .@y.values
```