|  |
| --- |
| “MNIST” |
| Charlotte Bahr |
| 6/27/2020 |
|  |

library(kerasR) #DNN  
library(tensorflow)#DNN  
library(data.table)  
library(dataPreparation)  
library(e1071) #Naive Bayes  
library(randomForest) #random Forest  
library(class) #k-Nearest Neighbor   
library(kernlab) #Support Vector Machine  
library(mlbench)#contains Glass data  
library(keras) #contains MNIST data  
library(caret)  
library(mltest)  
library(dplyr) #wg. select  
use\_session\_with\_seed(9876)

## Set session seed to 9876 (disabled GPU, CPU parallelism)

## DATA PRE PROCESSING  
  
mnist <- dataset\_mnist()  
  
set.seed(1234)  
mnist <- as.data.frame(mnist$train)  
nzv <- as.matrix(nearZeroVar(mnist, saveMetrics= FALSE))  
mnist <- mnist[,-(as.matrix(nearZeroVar(mnist, saveMetrics= FALSE)))]  
  
standard.features <- scale(mnist[,1:249])  
mnist <- cbind(standard.features, mnist[250])  
  
set.seed(1234)  
train\_index <- sample(1:nrow(mnist), 0.7\*nrow(mnist))  
test\_index <- setdiff(1:nrow(mnist), train\_index)  
  
Train <- mnist[train\_index,]  
Test <- mnist[test\_index,]  
  
True\_Label <- Test$y  
  
# Reshape & rescale & One\_hot  
  
X\_train <- Train %>%   
 select(-y)%>%   
 as.matrix()  
  
Y\_train <- to\_categorical(Train$y)  
  
X\_test <- Test %>%   
 select(-y)%>%   
 as.matrix()  
  
Y\_test <- to\_categorical(Test$y)  
  
##MODELS   
  
#k-Nearest Neighbor  
  
pc <- proc.time()  
model\_KNN <- knn(Train[1:249], Test[1:249], as.factor(Train$y), k=205) ##sqrt42000 =204,939  
print(proc.time() - pc)

## user system elapsed   
## 547.139 2.347 551.908

#Naive Bayes  
  
pc <- proc.time()   
model\_NB <- naiveBayes(as.factor(Train$y) ~. , Train[1:249])  
print(proc.time() - pc)

## user system elapsed   
## 1.391 0.899 2.501

#Random Forest  
  
pc <- proc.time()   
model\_RF <- randomForest(as.factor(Train$y) ~. , Train[1:249])  
print(proc.time() - pc)

## user system elapsed   
## 952.880 7.792 965.963

#Support Vector Machine  
   
pc <- proc.time()   
model\_SVM <- ksvm(Train$y ~. , Train[1:249], type = "C-svc", C = 1, kernel = "rbfdot" )  
print(proc.time() - pc)

## user system elapsed   
## 366.193 34.369 409.329

#Deep Neural Network  
  
pc <- proc.time()   
  
model\_DNN <- Sequential()  
  
model\_DNN$add(Dense(units=250, input\_shape = dim(X\_train)[2]))  
model\_DNN$add(LeakyReLU())  
model\_DNN$add(Dropout(0.4))  
  
model\_DNN$add(Dense(units=250))  
model\_DNN$add(LeakyReLU())  
model\_DNN$add(Dropout(0.3))  
  
model\_DNN$add(Dense(units=250))  
model\_DNN$add(LeakyReLU())  
model\_DNN$add(Dropout(0.2))  
  
model\_DNN$add(Dense(units=250))  
model\_DNN$add(LeakyReLU())  
model\_DNN$add(Dropout(0.1))  
  
model\_DNN$add(Dense(10))  
model\_DNN$add(Activation("softmax"))  
  
# compile  
keras\_compile(model\_DNN, loss ="categorical\_crossentropy", optimizer = RMSprop(), metrics = "accuracy")  
keras\_fit(model\_DNN, as.matrix(X\_train), Y\_train, batch\_size = 128, epochs = 32, verbose= 1, validation\_split = 0.2)  
  
print(proc.time() - pc)

## user system elapsed   
## 99.624 6.517 109.609

##EVALUATION METRICS  
  
#predictions  
set.seed(8912)  
  
pred\_KNN <- #has no prediction value  
  
pred\_NB <- as.factor(predict(model\_NB, Test))  
  
pred\_RF <- as.factor(predict(model\_RF, Test))  
  
pred\_SVM <- as.factor(predict(model\_SVM, Test))  
  
pred\_DNN <- as.factor(keras\_predict\_classes(model\_DNN, as.matrix(X\_test)))  
  
  
CF\_KNN <- table(model\_KNN, True\_Label)  
   
CF\_NB <- table(pred\_NB, True\_Label)  
   
CF\_RF <- table(pred\_RF, True\_Label)  
   
CF\_SVM <- table(pred\_SVM, True\_Label)  
  
CF\_DNN <- table(pred\_DNN, True\_Label)  
  
  
print(CF\_KNN)

## True\_Label  
## model\_KNN 0 1 2 3 4 5 6 7 8 9  
## 0 1725 0 27 7 2 12 17 3 7 9  
## 1 2 1970 108 36 53 37 18 80 98 22  
## 2 2 6 1530 19 1 0 0 3 8 3  
## 3 0 2 10 1680 0 34 0 1 79 29  
## 4 2 3 14 1 1587 4 1 6 10 16  
## 5 8 1 4 31 1 1465 7 1 43 3  
## 6 36 2 15 15 21 23 1723 0 22 1  
## 7 5 1 56 32 11 10 1 1685 20 71  
## 8 0 0 7 13 0 0 0 0 1435 0  
## 9 3 1 9 20 112 30 0 22 57 1693

print(CF\_NB)

## True\_Label  
## pred\_NB 0 1 2 3 4 5 6 7 8 9  
## 0 1641 0 19 12 2 47 16 21 5 11  
## 1 0 1892 66 58 2 31 20 34 110 27  
## 2 16 36 1395 93 17 60 36 52 42 15  
## 3 3 3 15 1351 1 127 0 3 31 19  
## 4 5 2 21 4 1409 27 25 55 14 134  
## 5 55 24 14 123 16 1127 33 11 104 25  
## 6 19 9 102 28 43 49 1597 1 12 12  
## 7 1 1 22 5 3 19 0 1491 4 34  
## 8 38 14 119 116 41 50 40 27 1407 69  
## 9 5 5 7 64 254 78 0 106 50 1501

print(CF\_RF)

## True\_Label  
## pred\_RF 0 1 2 3 4 5 6 7 8 9  
## 0 1752 0 12 3 3 8 11 1 3 6  
## 1 0 1958 4 3 4 5 1 6 9 4  
## 2 5 8 1703 24 2 2 2 26 12 5  
## 3 0 5 8 1760 0 18 0 1 11 27  
## 4 3 7 10 2 1730 2 1 14 3 17  
## 5 3 0 2 27 0 1556 8 0 14 9  
## 6 7 3 6 5 10 10 1740 0 13 0  
## 7 1 0 21 7 5 3 0 1737 1 15  
## 8 11 3 11 16 1 7 4 5 1698 19  
## 9 1 2 3 7 33 4 0 11 15 1745

print(CF\_SVM)

## True\_Label  
## pred\_SVM 0 1 2 3 4 5 6 7 8 9  
## 0 1762 0 10 0 3 9 9 2 3 6  
## 1 0 1961 4 7 3 8 1 7 10 3  
## 2 5 10 1714 19 5 2 3 17 11 5  
## 3 0 3 5 1766 0 18 0 2 13 14  
## 4 4 6 8 0 1745 4 3 11 5 24  
## 5 4 1 4 26 0 1558 10 3 10 11  
## 6 3 1 5 4 7 9 1738 1 9 0  
## 7 0 1 18 7 8 1 0 1747 4 22  
## 8 1 1 9 20 1 1 3 3 1705 10  
## 9 4 2 3 5 16 5 0 8 9 1752

print(CF\_DNN)

## True\_Label  
## pred\_DNN 0 1 2 3 4 5 6 7 8 9  
## 0 1755 1 9 4 2 7 8 6 5 7  
## 1 0 1968 2 1 9 3 1 15 18 6  
## 2 0 8 1721 36 1 1 7 26 5 3  
## 3 1 1 5 1761 0 11 0 0 7 12  
## 4 5 2 5 0 1723 0 6 3 3 22  
## 5 6 0 3 25 1 1573 13 0 24 15  
## 6 10 2 2 3 6 11 1727 1 13 0  
## 7 2 2 11 8 7 2 0 1744 5 32  
## 8 2 2 14 10 1 2 5 1 1687 4  
## 9 2 0 8 6 38 5 0 5 12 1746

#metrics  
  
ml\_test\_KNN <- ml\_test(model\_KNN, True\_Label, output.as.table = FALSE)  
  
ml\_test\_NB <- ml\_test(pred\_NB, True\_Label, output.as.table = FALSE)  
  
ml\_test\_RF <- ml\_test(pred\_RF, True\_Label, output.as.table = FALSE)  
  
ml\_test\_SVM <- ml\_test(pred\_SVM, True\_Label, output.as.table = FALSE)  
  
ml\_test\_DNN <- ml\_test(pred\_DNN, True\_Label, output.as.table = FALSE)  
  
#Macro Average Accuracy  
  
MAvA\_KNN <- print((sum(ml\_test\_KNN$balanced.accuracy, na.rm = TRUE))/10)

## [1] 0.9526622

MAvA\_NB <- print((sum(ml\_test\_NB$balanced.accuracy, na.rm = TRUE))/10)

## [1] 0.8987109

MAvA\_RF <- print((sum(ml\_test\_RF$balanced.accuracy, na.rm = TRUE))/10)

## [1] 0.9807252

MAvA\_SVM <- print((sum(ml\_test\_SVM$balanced.accuracy, na.rm = TRUE))/10)

## [1] 0.982866

MAvA\_DNN <- print((sum(ml\_test\_DNN$balanced.accuracy, na.rm = TRUE))/10)

## [1] 0.9815537

#Macro Average F1   
  
MAvF1\_KNN <- print((sum(ml\_test\_KNN$F1, na.rm = TRUE))/10)

## [1] 0.916716

MAvF1\_NB <- print((sum(ml\_test\_NB$F1, na.rm = TRUE))/10)

## [1] 0.8210583

MAvF1\_RF <- print((sum(ml\_test\_RF$F1, na.rm = TRUE))/10)

## [1] 0.965321

MAvF1\_SVM <- print((sum(ml\_test\_SVM$F1, na.rm = TRUE))/10)

## [1] 0.9691426

MAvF1\_DNN <- print((sum(ml\_test\_DNN$F1, na.rm = TRUE))/10)

## [1] 0.9667566

#MAvMCC  
  
MAvMCC\_KNN <- print((sum(ml\_test\_KNN$MCC, na.rm = TRUE))/10)

## [1] 0.908268

MAvMCC\_NB <- print((sum(ml\_test\_NB$MCC, na.rm = TRUE))/10)

## [1] 0.7990532

MAvMCC\_RF <- print((sum(ml\_test\_RF$MCC, na.rm = TRUE))/10)

## [1] 0.9613825

MAvMCC\_SVM <- print((sum(ml\_test\_SVM$MCC, na.rm = TRUE))/10)

## [1] 0.9656671

MAvMCC\_DNN <- print((sum(ml\_test\_DNN$MCC, na.rm = TRUE))/10)

## [1] 0.9630304

#MAvGeometricMean  
  
MAvGM\_KNN <- print((sum(ml\_test\_KNN$geometric.mean, na.rm = TRUE))/10)

## [1] 0.9514375

MAvGM\_NB <- print((sum(ml\_test\_NB$geometric.mean, na.rm = TRUE))/10)

## [1] 0.8943387

MAvGM\_RF <- print((sum(ml\_test\_RF$geometric.mean, na.rm = TRUE))/10)

## [1] 0.9805821

MAvGM\_SVM <- print((sum(ml\_test\_SVM$geometric.mean, na.rm = TRUE))/10)

## [1] 0.9827485

MAvGM\_DNN <- print((sum(ml\_test\_DNN$geometric.mean, na.rm = TRUE))/10)

## [1] 0.9814153