Trabalho de Aplicação

Yugo Oyama NUSP: 9297784

14/11/2021

Introdução

Este trabalho consiste na aplicação de técnicas estatísticas abordadas no curso de Estatística em Altas Dimensões - 2021 em dois bancos de dados que serão explicados detalhadamente mais a frente.

Dados de digitos manuscritos

```
library(dplyr)
library(glmnet) # tem as funcoes para lasso, ridge e elasticnet
library(keras)
library(randomForest)
```

```
load("dados_mnist.rdata")
load("dados_mnist_teste.rdata")
```

O banco de dados é composto por:
x_treino: matriz com as 60000 imagens do conjunto de treino;
y_treino: vetor com os reais valores dos dígitos escritos nas imagens do conjunto de treino (etiquetas);

Cada linha da matriz x_treino é uma imagem. Aqui, cada imagem está representada como um vetor de dimensao 1x784. Cada coluna indica o tom de cinza do respectivo pixel da imagem (entre 0 - preto e 255 - branco).

Objetivo

O objetivo desta análise é propor um modelo que consiga realizar boas predições para a qual número corresponde cada uma das imagens.

Técnicas

Para esse projeto serão usados modelos lineares com regularização, modelos baseados em árvores, e redes neurais.

Para comparar os modelos entre si,será utilizada a acurácia e ao final de cada método testado, será adicionado o resultado a uma tabela comparativa.

Por motivos de tempo de processamento, apenas o modelo escolhido de cada categoria será apresentado no relatório.

Parâmetros

Para o ajuste do modelo, foi definido que o conjunto de treino seria dividido em conjunto de treino e validação na proporção 7,5:1,5, ou seja, 75% do conjunto originalmente de treino foi usado para validação. Foi definida também uma semente por questão de reprodutibilidade (12345).

Lasso

Para criar um modelo de regressão com a penalização lasso, foi utilizada a biblioteca glmnet. Com ela, dentro da amostra de treino, foi ajustado o modelo de classificação com penalização lasso utizando validação cruzada. Em seguida, foi testado o modelo obtido no conjunto de validação e calculado o respectivo erro dentro e fora

Para o ajuste do modelo, inicialmente testou-se os definir para a função os valores (0.01, 0.1, 1, 2, 10, 25, 50, 75, 100) e em seguida testou-se definir que o modelo escolhesse 30 lambdas diferentes.

Existem dois lambdas que são popularmentes usados: o que minimiza o erro gerado pela validação cruzada e o no qual o erro não ultrapassa um desvio padrão do melhor modelo. Com isso, foram testados modelos com cada um dos lambdas e calculados os erros dentro e fora respectivos de cada um.

O modelo cujos valores de lambda foram escolhidos pelo cv.glmnet apresentou resultados significativamente melhores.

O cálculo de previsão foi realizado diretamente do modelo resultante da validação cruzada conforme recomendado na documentação do pacote glmnet por questão de convergência e otimização.

```
# y_treino <- factor(y_treino)
db_digitos <- data.frame(y_treino=y_treino,x_treino)

set.seed(12345)
X <- model.matrix(db_digitos[,1] ~ .,
                  data = db_digitos[,-1])[,-1] # X deve ser uma matrix sem intercepto
```

```
# separando 75% dos dados para treino
ids <- sample(nrow(db_digitos), size = .75*nrow(db_digitos), replace = FALSE)
```

```
cv_lasso <- cv.glmnet(X[ids,], as.factor(db_digitos$y_treino[ids]), family = "multinomial", alpha = 1,
                      type.measure = "class", trace.it = TRUE, nlambda = 30, maxit = 10000, nfolds = 10)
```

```
## Training
##   |                                                                      |
## Warning: from glmnet Fortran code (error code -23); Convergence for 23th lambda
## value not reached after maxit=10000 iterations; solutions for larger lambdas
## returned
##   |                                                                      |==================
## Fold: 1/10
##   |                                                                      |
## Warning: from glmnet Fortran code (error code -23); Convergence for 23th lambda
## value not reached after maxit=10000 iterations; solutions for larger lambdas
## returned
##   |                                                                      |==================
## Fold: 2/10
##   |                                                                      |
## Warning: from glmnet Fortran code (error code -23); Convergence for 23th lambda
## value not reached after maxit=10000 iterations; solutions for larger lambdas
## returned
##   |                                                                      |==================
## Fold: 3/10
##   |                                                                      |
## Warning: from glmnet Fortran code (error code -23); Convergence for 23th lambda
## value not reached after maxit=10000 iterations; solutions for larger lambdas
```

```
## returned
##   |                                                                      |==================
## Fold: 4/10
##   |                                                                      |
## Warning: from glmnet Fortran code (error code -23); Convergence for 23th lambda
## value not reached after maxit=10000 iterations; solutions for larger lambdas
## returned
##   |                                                                      |==================
## Fold: 5/10
##   |                                                                      |
## Warning: from glmnet Fortran code (error code -23); Convergence for 23th lambda
## value not reached after maxit=10000 iterations; solutions for larger lambdas
## returned
##   |                                                                      |==================
## Fold: 6/10
##   |                                                                      |
## Warning: from glmnet Fortran code (error code -27); Convergence for 27th lambda
## value not reached after maxit=10000 iterations; solutions for larger lambdas
## returned
##   |                                                                      |==================
## Fold: 7/10
##   |                                                                      |
## Warning: from glmnet Fortran code (error code -23); Convergence for 23th lambda
## value not reached after maxit=10000 iterations; solutions for larger lambdas
## returned
##   |                                                                      |==================
## Fold: 8/10
##   |                                                                      |
## Warning: from glmnet Fortran code (error code -23); Convergence for 23th lambda
## value not reached after maxit=10000 iterations; solutions for larger lambdas
## returned
##   |                                                                      |==================
## Fold: 9/10
##   |                                                                      |
## Warning: from glmnet Fortran code (error code -23); Convergence for 23th lambda
## value not reached after maxit=10000 iterations; solutions for larger lambdas
## returned
##   |                                                                      |==================
## Fold: 10/10
##   |                                                                      |
## Warning: from glmnet Fortran code (error code -27); Convergence for 27th lambda
## value not reached after maxit=10000 iterations; solutions for larger lambdas
## returned
##   |                                                                      |==================
```

```
cv_lasso$lambda.min
```

```
## [1] 0.000291764
```

```
cv_lasso$lambda.1se
```

```
## [1] 0.0004008323
```

```
# saveRDS(cv_lasso, "mnist_lasso2.rds")
```

```
# lasso <- glmnet(X[ids,], as.factor(db_digitos$y_treino[ids]), alpha = 1, lambda = 0.05, family = "mul
```

```
saveRDS(cv_lasso, "mnist_lasso.rds")
```

```
cv_lasso <- readRDS("mnist_lasso.rds")
```

```
cv_lasso <-readRDS("mnist_lasso.rds")

# cv_lasso2 <-readRDS("mnist_lasso2.rds")
```

```
y_lasso_dentro <- predict(cv_lasso, newx = X[ids,],
                          s = cv_lasso$lambda.min, type = "class") # valor predito dentro da amostra

y_lasso_dentro1.2 <- predict(cv_lasso, newx = X[ids,],
                          s = cv_lasso$lambda.1se, type = "class") # valor predito dentro da amostra

# y_lasso_dentro2 <- predict(cv_lasso2, newx = X[ids,],
#                            s = cv_lasso2$lambda.min, type = "class") # valor predito dentro da amostra
#
# y_lasso_dentro2.2 <- predict(cv_lasso2, newx = X[ids,],
#                            s = cv_lasso2$lambda.1se, type = "class") # valor predito dentro da amostra
```

```
# comparativo: observado vs predito (dentro da amostra)
(tabela_tr <- table(predito = y_lasso_dentro, observado = y_treino[ids]))
```

```
#
y_lasso_fora <- predict(cv_lasso, newx = X[-ids,],
                        s = cv_lasso$lambda.min, type = "class") # valor predito fora da amostra

y_lasso_fora1.2 <- predict(cv_lasso, newx = X[-ids,],
                        s = cv_lasso$lambda.1se, type = "class") # valor

# y_lasso_fora2 <- predict(cv_lasso, newx = X[-ids,],
#                          s = cv_lasso2$lambda.min, type = "class") # valor predito fora da amostra
#
# y_lasso_fora2.2 <- predict(cv_lasso, newx = X[-ids,],
#                          s = cv_lasso2$lambda.1se, type = "class") # valor
```

```
lasso_ac_dentro<- mean(y_lasso_dentro==y_treino[ids])
lasso_ac_dentro1.2<- mean(y_lasso_dentro1.2==y_treino[ids])
lasso_ac_fora<- mean(y_lasso_fora==y_treino[-ids])
lasso_ac_fora1.2<- mean(y_lasso_fora1.2==y_treino[-ids])

# lasso_ac_dentro2<- mean(y_lasso_dentro2==y_treino[ids])
# lasso_ac_dentro2.2<- mean(y_lasso_dentro2.2==y_treino[ids])
# lasso_ac_fora2<- mean(y_lasso_fora2==y_treino[-ids])
# lasso_ac_fora2.2<- mean(y_lasso_fora2.2==y_treino[-ids])
```

```
resultados <- data.frame("Lasso - Lambda minimo",lasso_ac_dentro, lasso_ac_fora)
```

```
names(resultados) <- c("modelo", "acuracia_dentro", "acuracia_fora")

resultados <- rbind(resultados,data.frame(modelo="Lasso - Lambda 1 desvio padrao",acuracia_dentro=lasso_

# resultados <- rbind(resultados,data.frame(modelo="Lasso2 - Lambda minimo padrao",acuracia_dentro=lass
#
# resultados <- rbind(resultados,data.frame(modelo="Lasso2 - Lambda 1 desvio padrao",acuracia_dentro=la

resultados <- resultados %>% mutate(across(where(is.numeric),round,4))
resultados
```

```
##                           modelo acuracia_dentro acuracia_fora
## 1          Lasso - Lambda minimo          0.9364        0.9219
## 2 Lasso - Lambda 1 desvio padrao          0.9342        0.9207
```

Modelo de Árvores

Foi escolhido o modelo de Floresta Aleatória já que dessa forma existe variabilidade na escolha das variáveis explicativas e na escolha das observações para a construção de cada árvore. Essa variabilidade por sua vez pode levar a uma melhor predição no conjunto de validação e no de teste.

Para realizar o ajuste do modelo, foi utilizado a função randomForest do pacote randomForest.

```
# library(tree) # funcoes para estimar arvore de reg/class
# library(randomForest) # funcoes para estimar floresta aleatoria
```

```
modelo_rf <- randomForest(factor(db_digitos$y_treino[ids]) ~ .,
                    data = db_digitos[ids, ], do.trace=TRUE)
```

```
saveRDS(modelo_rf, "mnist_rf.rds")
```

```
modelo_rf <- readRDS("mnist_rf.rds")
```

```
# importancia das variaveis
importance(modelo_rf)
```

```
##      MeanDecreaseGini
## X1       0.000000e+00
## X2       0.000000e+00
## X3       0.000000e+00
## X4       0.000000e+00
## X5       0.000000e+00
## X6       0.000000e+00
## X7       0.000000e+00
## X8       0.000000e+00
## X9       0.000000e+00
## X10      0.000000e+00
## X11      0.000000e+00
## X12      0.000000e+00
## X13      0.000000e+00
## X14      0.000000e+00
## X15      0.000000e+00
## X16      0.000000e+00
## X17      0.000000e+00
## X18      0.000000e+00
```

```
## X19      0.000000e+00
## X20      0.000000e+00
## X21      0.000000e+00
## X22      0.000000e+00
## X23      0.000000e+00
## X24      0.000000e+00
## X25      0.000000e+00
## X26      0.000000e+00
## X27      0.000000e+00
## X28      0.000000e+00
## X29      0.000000e+00
## X30      0.000000e+00
## X31      0.000000e+00
## X32      0.000000e+00
## X33      0.000000e+00
## X34      0.000000e+00
## X35      6.111953e-02
## X36      1.185463e-01
## X37      1.099022e-01
## X38      1.006825e-01
## X39      1.059071e-01
## X40      2.504465e-01
## X41      1.345936e-01
## X42      1.033842e-01
## X43      2.085827e-01
## X44      1.896821e-01
## X45      1.904237e-01
## X46      9.879764e-02
## X47      3.666359e-02
## X48      2.868400e-02
## X49      5.958474e-02
## X50      1.111489e-02
## X51      3.778514e-02
## X52      3.789184e-02
## X53      0.000000e+00
## X54      0.000000e+00
## X55      0.000000e+00
## X56      0.000000e+00
## X57      0.000000e+00
## X58      0.000000e+00
## X59      0.000000e+00
## X60      0.000000e+00
## X61      0.000000e+00
## X62      1.507795e-02
## X63      3.036132e-01
## X64      4.734349e-01
## X65      8.338833e-01
## X66      1.412807e+00
## X67      2.319399e+00
## X68      5.385482e+00
## X69      7.566225e+00
## X70      5.774430e+00
## X71      5.968055e+00
## X72      8.441186e+00
```

```
## X73      7.970452e+00
## X74      3.464156e+00
## X75      2.715383e+00
## X76      2.194760e+00
## X77      1.402828e+00
## X78      5.164912e-01
## X79      3.599449e-01
## X80      7.045877e-02
## X81      2.608218e-02
## X82      2.321113e-02
## X83      0.000000e+00
## X84      0.000000e+00
## X85      0.000000e+00
## X86      0.000000e+00
## X87      1.530112e-02
## X88      1.965101e-02
## X89      1.915741e-02
## X90      9.602404e-02
## X91      5.652941e-01
## X92      1.013407e+00
## X93      2.441087e+00
## X94      5.706797e+00
## X95      1.037282e+01
## X96      2.120749e+01
## X97      3.615667e+01
## X98      2.733036e+01
## X99      5.384853e+01
## X100     9.738141e+01
## X101     8.938199e+01
## X102     6.694869e+01
## X103     6.051091e+01
## X104     3.304691e+01
## X105     1.505270e+01
## X106     5.100817e+00
## X107     2.127308e+00
## X108     1.173721e+00
## X109     4.062867e-01
## X110     1.493817e-01
## X111     1.334901e-02
## X112     0.000000e+00
## X113     0.000000e+00
## X114     0.000000e+00
## X115     7.560793e-03
## X116     7.075844e-02
## X117     1.701466e-01
## X118     4.770649e-01
## X119     1.328099e+00
## X120     3.304680e+00
## X121     6.193952e+00
## X122     1.242058e+01
## X123     2.341745e+01
## X124     5.148528e+01
## X125     6.373456e+01
## X126     8.220915e+01
```

```
## X127     9.434970e+01
## X128     7.822787e+01
## X129     5.908920e+01
## X130     4.798479e+01
## X131     3.668065e+01
## X132     2.390646e+01
## X133     1.466141e+01
## X134     9.479257e+00
## X135     5.684948e+00
## X136     3.263616e+00
## X137     1.746651e+00
## X138     7.482315e-01
## X139     7.314731e-02
## X140     4.000000e-03
## X141     0.000000e+00
## X142     0.000000e+00
## X143     1.576926e-02
## X144     1.330910e-01
## X145     4.714038e-01
## X146     1.661056e+00
## X147     4.536682e+00
## X148     9.279095e+00
## X149     2.477720e+01
## X150     4.899741e+01
## X151     7.379006e+01
## X152     1.181849e+02
## X153     1.414225e+02
## X154     1.926535e+02
## X155     2.021927e+02
## X156     2.538572e+02
## X157     2.012435e+02
## X158     1.235832e+02
## X159     8.894687e+01
## X160     6.008918e+01
## X161     4.275389e+01
## X162     2.995023e+01
## X163     2.012583e+01
## X164     1.175629e+01
## X165     4.407392e+00
## X166     1.230532e+00
## X167     2.674388e-01
## X168     0.000000e+00
## X169     0.000000e+00
## X170     7.614353e-03
## X171     6.049236e-02
## X172     2.916597e-01
## X173     1.346317e+00
## X174     3.817923e+00
## X175     9.548580e+00
## X176     2.025751e+01
## X177     4.417060e+01
## X178     7.464188e+01
## X179     1.049840e+02
## X180     1.102717e+02
```

```
## X181     1.006956e+02
## X182     1.183812e+02
## X183     1.800335e+02
## X184     1.786172e+02
## X185     1.361889e+02
## X186     1.151820e+02
## X187     8.432987e+01
## X188     6.297584e+01
## X189     5.989872e+01
## X190     5.562101e+01
## X191     4.942544e+01
## X192     4.183945e+01
## X193     1.556047e+01
## X194     2.431640e+00
## X195     3.136632e-01
## X196     6.296663e-02
## X197     0.000000e+00
## X198     7.670451e-03
## X199     1.253989e-01
## X200     5.741777e-01
## X201     2.733979e+00
## X202     6.523497e+00
## X203     1.308151e+01
## X204     2.488381e+01
## X205     4.485532e+01
## X206     6.576518e+01
## X207     8.454804e+01
## X208     9.847540e+01
## X209     1.067638e+02
## X210     1.651763e+02
## X211     2.488553e+02
## X212     2.405969e+02
## X213     1.782456e+02
## X214     1.177952e+02
## X215     9.926791e+01
## X216     8.796534e+01
## X217     7.737220e+01
## X218     6.010676e+01
## X219     5.769248e+01
## X220     5.027043e+01
## X221     2.991142e+01
## X222     4.665322e+00
## X223     6.562069e-01
## X224     5.531230e-02
## X225     0.000000e+00
## X226     2.540912e-02
## X227     2.205290e-01
## X228     8.596213e-01
## X229     2.992779e+00
## X230     8.706666e+00
## X231     1.497999e+01
## X232     2.791711e+01
## X233     4.321109e+01
## X234     7.374739e+01
```

```
## X235     1.244857e+02
## X236     1.273557e+02
## X237     1.362510e+02
## X238     1.743440e+02
## X239     2.167784e+02
## X240     2.107390e+02
## X241     1.615272e+02
## X242     1.355720e+02
## X243     1.274515e+02
## X244     1.144853e+02
## X245     9.255507e+01
## X246     6.141180e+01
## X247     4.949331e+01
## X248     4.147010e+01
## X249     3.034356e+01
## X250     7.730883e+00
## X251     6.680262e-01
## X252     5.975411e-02
## X253     3.657895e-03
## X254     6.169980e-02
## X255     1.685333e-01
## X256     1.095970e+00
## X257     3.714256e+00
## X258     1.051639e+01
## X259     2.051740e+01
## X260     3.321600e+01
## X261     4.632675e+01
## X262     8.078231e+01
## X263     1.457592e+02
## X264     1.904544e+02
## X265     1.323073e+02
## X266     1.274312e+02
## X267     1.538710e+02
## X268     1.543570e+02
## X269     1.415361e+02
## X270     1.547113e+02
## X271     1.707948e+02
## X272     1.323796e+02
## X273     1.010798e+02
## X274     6.873193e+01
## X275     4.176167e+01
## X276     3.346080e+01
## X277     2.123329e+01
## X278     5.451784e+00
## X279     8.340045e-01
## X280     5.969962e-02
## X281     0.000000e+00
## X282     3.794843e-02
## X283     2.007465e-01
## X284     1.159582e+00
## X285     5.155019e+00
## X286     1.107163e+01
## X287     1.797697e+01
## X288     3.347672e+01
```

```
## X289     6.515074e+01
## X290     1.277073e+02
## X291     2.360605e+02
## X292     2.002548e+02
## X293     1.294223e+02
## X294     1.127731e+02
## X295     1.310901e+02
## X296     1.494066e+02
## X297     1.535611e+02
## X298     1.411579e+02
## X299     1.745511e+02
## X300     1.464046e+02
## X301     1.066492e+02
## X302     8.689383e+01
## X303     5.306078e+01
## X304     1.980946e+01
## X305     1.027970e+01
## X306     2.633612e+00
## X307     7.365758e-01
## X308     9.616852e-02
## X309     0.000000e+00
## X310     7.435689e-02
## X311     2.227224e-01
## X312     1.224091e+00
## X313     4.076350e+00
## X314     8.556900e+00
## X315     2.650512e+01
## X316     5.091652e+01
## X317     8.829342e+01
## X318     1.662487e+02
## X319     2.302909e+02
## X320     2.108358e+02
## X321     1.574226e+02
## X322     1.264331e+02
## X323     1.876633e+02
## X324     1.790496e+02
## X325     1.442722e+02
## X326     1.345953e+02
## X327     1.732680e+02
## X328     1.285235e+02
## X329     9.124966e+01
## X330     8.194995e+01
## X331     6.610844e+01
## X332     2.652712e+01
## X333     5.150205e+00
## X334     1.438647e+00
## X335     4.613431e-01
## X336     9.024783e-02
## X337     0.000000e+00
## X338     3.473324e-02
## X339     2.225307e-01
## X340     7.090039e-01
## X341     3.800863e+00
## X342     1.228117e+01
```

```
## X343     3.486340e+01
## X344     5.796700e+01
## X345     1.103776e+02
## X346     1.795807e+02
## X347     2.086881e+02
## X348     2.453260e+02
## X349     1.895688e+02
## X350     1.861266e+02
## X351     3.564352e+02
## X352     2.091081e+02
## X353     1.418120e+02
## X354     1.559741e+02
## X355     1.595649e+02
## X356     9.355862e+01
## X357     6.794407e+01
## X358     6.993604e+01
## X359     8.740434e+01
## X360     2.454809e+01
## X361     5.203348e+00
## X362     9.605158e-01
## X363     2.642997e-01
## X364     3.927273e-03
## X365     0.000000e+00
## X366     2.347541e-02
## X367     7.671738e-02
## X368     7.216151e-01
## X369     3.485530e+00
## X370     1.400739e+01
## X371     4.394409e+01
## X372     7.205718e+01
## X373     1.325901e+02
## X374     2.087905e+02
## X375     2.251068e+02
## X376     2.293989e+02
## X377     1.892721e+02
## X378     2.832693e+02
## X379     3.751617e+02
## X380     1.816908e+02
## X381     1.681602e+02
## X382     2.050391e+02
## X383     1.465712e+02
## X384     6.814415e+01
## X385     5.120054e+01
## X386     7.365516e+01
## X387     8.453232e+01
## X388     3.772851e+01
## X389     6.310268e+00
## X390     6.817779e-01
## X391     2.554783e-01
## X392     1.167895e-02
## X393     3.733333e-03
## X394     2.266249e-02
## X395     3.112363e-02
## X396     6.486503e-01
```

```
## X397     3.701400e+00
## X398     1.611448e+01
## X399     5.146939e+01
## X400     9.587152e+01
## X401     1.522156e+02
## X402     1.956144e+02
## X403     2.104551e+02
## X404     1.913255e+02
## X405     1.821985e+02
## X406     3.007418e+02
## X407     3.524710e+02
## X408     1.796140e+02
## X409     1.647816e+02
## X410     3.239503e+02
## X411     1.499556e+02
## X412     6.137567e+01
## X413     4.872513e+01
## X414     6.541487e+01
## X415     5.716851e+01
## X416     2.767412e+01
## X417     6.384826e+00
## X418     1.279139e+00
## X419     3.815062e-01
## X420     9.383824e-03
## X421     9.612745e-03
## X422     0.000000e+00
## X423     1.558498e-01
## X424     8.765654e-01
## X425     3.544971e+00
## X426     1.867575e+01
## X427     5.633521e+01
## X428     1.002553e+02
## X429     1.598433e+02
## X430     2.008375e+02
## X431     2.002777e+02
## X432     1.720775e+02
## X433     1.805943e+02
## X434     3.179197e+02
## X435     2.274304e+02
## X436     1.880966e+02
## X437     1.695039e+02
## X438     2.431751e+02
## X439     1.331136e+02
## X440     7.400178e+01
## X441     6.451866e+01
## X442     5.642310e+01
## X443     4.242587e+01
## X444     2.100473e+01
## X445     6.453578e+00
## X446     1.870653e+00
## X447     4.581867e-01
## X448     3.511587e-02
## X449     0.000000e+00
## X450     0.000000e+00
```

```
## X451     1.497397e-01
## X452     1.107906e+00
## X453     3.927593e+00
## X454     2.239833e+01
## X455     5.524068e+01
## X456     1.317985e+02
## X457     1.569833e+02
## X458     1.864439e+02
## X459     1.605585e+02
## X460     1.891785e+02
## X461     2.483983e+02
## X462     2.778000e+02
## X463     2.500727e+02
## X464     1.403540e+02
## X465     1.286152e+02
## X466     1.378826e+02
## X467     1.065702e+02
## X468     6.607141e+01
## X469     7.476051e+01
## X470     5.873350e+01
## X471     3.223461e+01
## X472     1.613779e+01
## X473     6.758040e+00
## X474     2.917660e+00
## X475     6.284957e-01
## X476     1.344394e-01
## X477     0.000000e+00
## X478     0.000000e+00
## X479     2.024388e-01
## X480     1.812163e+00
## X481     4.681203e+00
## X482     2.466947e+01
## X483     4.656713e+01
## X484     1.035846e+02
## X485     1.323932e+02
## X486     1.616375e+02
## X487     1.919249e+02
## X488     2.249626e+02
## X489     2.478389e+02
## X490     2.636462e+02
## X491     1.542934e+02
## X492     1.125071e+02
## X493     8.024415e+01
## X494     1.087165e+02
## X495     8.981336e+01
## X496     8.325761e+01
## X497     8.629137e+01
## X498     3.648001e+01
## X499     2.859142e+01
## X500     1.438281e+01
## X501     8.002410e+00
## X502     2.474343e+00
## X503     5.843916e-01
## X504     1.013034e-01
```

```
## X505     0.000000e+00
## X506     7.707781e-03
## X507     2.202490e-01
## X508     2.517274e+00
## X509     7.124876e+00
## X510     2.349450e+01
## X511     5.249464e+01
## X512     1.017201e+02
## X513     1.039833e+02
## X514     1.548723e+02
## X515     2.406233e+02
## X516     2.507946e+02
## X517     1.634886e+02
## X518     1.624999e+02
## X519     1.122002e+02
## X520     8.399980e+01
## X521     6.167356e+01
## X522     9.475944e+01
## X523     1.096234e+02
## X524     1.100049e+02
## X525     6.695418e+01
## X526     3.685772e+01
## X527     2.618516e+01
## X528     1.811466e+01
## X529     1.046531e+01
## X530     1.950703e+00
## X531     2.750888e-01
## X532     4.894099e-02
## X533     0.000000e+00
## X534     3.333333e-03
## X535     3.409150e-01
## X536     2.694686e+00
## X537     9.437808e+00
## X538     2.243183e+01
## X539     5.300704e+01
## X540     1.091724e+02
## X541     1.556297e+02
## X542     2.093174e+02
## X543     2.376596e+02
## X544     2.160197e+02
## X545     1.573639e+02
## X546     1.001969e+02
## X547     7.708992e+01
## X548     5.351687e+01
## X549     5.305361e+01
## X550     1.005170e+02
## X551     1.153702e+02
## X552     9.920638e+01
## X553     6.569342e+01
## X554     4.645593e+01
## X555     3.689804e+01
## X556     1.949382e+01
## X557     6.688031e+00
## X558     1.239184e+00
```

```
## X559     1.374345e-01
## X560     2.299606e-02
## X561     0.000000e+00
## X562     2.468169e-02
## X563     2.138889e-01
## X564     3.157537e+00
## X565     9.559276e+00
## X566     2.400302e+01
## X567     5.997742e+01
## X568     1.449967e+02
## X569     2.184059e+02
## X570     2.022752e+02
## X571     1.848074e+02
## X572     1.470583e+02
## X573     1.116647e+02
## X574     9.238070e+01
## X575     7.870244e+01
## X576     6.803346e+01
## X577     6.093553e+01
## X578     8.627672e+01
## X579     9.866981e+01
## X580     6.844917e+01
## X581     5.488345e+01
## X582     4.801476e+01
## X583     2.783183e+01
## X584     1.278697e+01
## X585     5.403460e+00
## X586     7.171433e-01
## X587     1.062216e-01
## X588     0.000000e+00
## X589     0.000000e+00
## X590     1.030505e-02
## X591     2.229765e-01
## X592     1.674432e+00
## X593     6.960483e+00
## X594     1.750300e+01
## X595     4.891566e+01
## X596     1.098529e+02
## X597     2.010986e+02
## X598     1.529904e+02
## X599     1.134348e+02
## X600     7.569295e+01
## X601     6.340482e+01
## X602     5.939493e+01
## X603     6.253821e+01
## X604     5.384896e+01
## X605     5.308305e+01
## X606     6.496422e+01
## X607     5.813634e+01
## X608     5.137862e+01
## X609     4.376097e+01
## X610     2.329966e+01
## X611     1.486701e+01
## X612     7.095221e+00
```

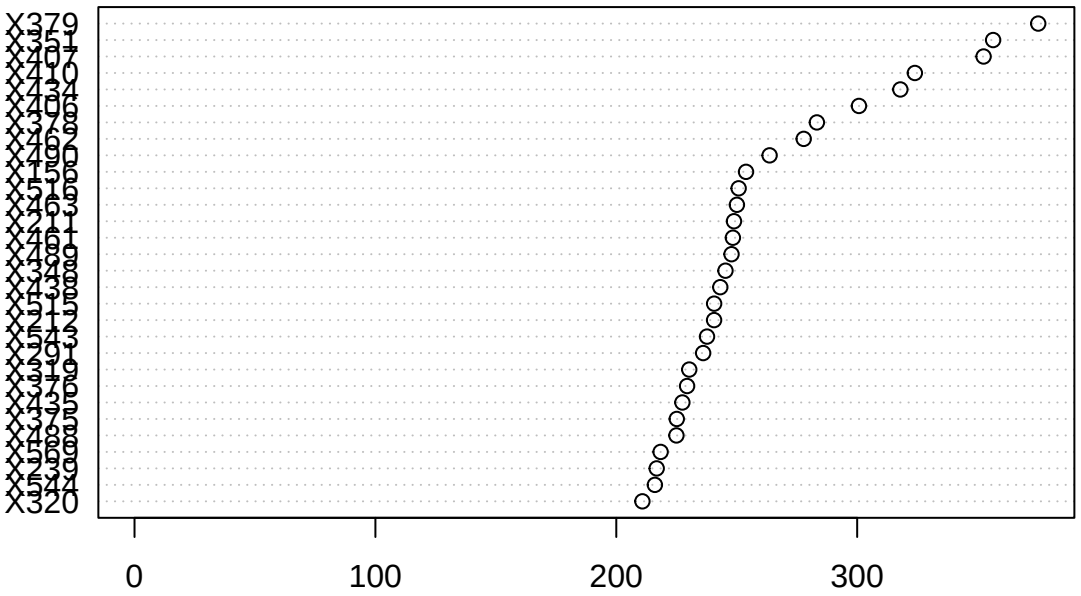
```
## X613     1.981041e+00
## X614     4.695954e-01
## X615     4.908601e-02
## X616     0.000000e+00
## X617     0.000000e+00
## X618     0.000000e+00
## X619     1.179868e-01
## X620     9.572887e-01
## X621     3.967169e+00
## X622     1.237234e+01
## X623     2.603275e+01
## X624     6.642363e+01
## X625     1.258268e+02
## X626     1.279487e+02
## X627     1.356846e+02
## X628     8.217889e+01
## X629     7.023946e+01
## X630     7.506438e+01
## X631     7.029935e+01
## X632     5.586235e+01
## X633     5.666535e+01
## X634     4.074707e+01
## X635     3.623344e+01
## X636     3.013024e+01
## X637     1.729567e+01
## X638     1.076127e+01
## X639     6.041976e+00
## X640     2.837313e+00
## X641     1.128224e+00
## X642     1.614604e-01
## X643     7.902992e-02
## X644     0.000000e+00
## X645     0.000000e+00
## X646     0.000000e+00
## X647     5.072528e-02
## X648     5.496788e-01
## X649     2.565469e+00
## X650     5.818276e+00
## X651     1.395248e+01
## X652     3.055773e+01
## X653     4.849329e+01
## X654     9.036601e+01
## X655     1.252103e+02
## X656     1.611586e+02
## X657     1.855209e+02
## X658     1.936900e+02
## X659     1.695658e+02
## X660     9.937464e+01
## X661     6.550911e+01
## X662     4.048146e+01
## X663     2.618623e+01
## X664     1.749849e+01
## X665     9.419431e+00
## X666     5.931544e+00
```

```
## X667     2.814692e+00
## X668     1.259287e+00
## X669     5.373609e-01
## X670     1.734377e-01
## X671     2.811479e-02
## X672     0.000000e+00
## X673     0.000000e+00
## X674     0.000000e+00
## X675     1.821945e-02
## X676     3.715760e-01
## X677     1.551203e+00
## X678     3.508319e+00
## X679     8.570663e+00
## X680     1.493184e+01
## X681     2.578078e+01
## X682     3.419692e+01
## X683     4.261959e+01
## X684     5.209594e+01
## X685     6.102198e+01
## X686     5.767474e+01
## X687     4.588250e+01
## X688     3.730196e+01
## X689     2.180342e+01
## X690     1.664042e+01
## X691     1.266461e+01
## X692     1.021681e+01
## X693     6.052772e+00
## X694     3.122264e+00
## X695     1.417018e+00
## X696     6.763068e-01
## X697     2.869847e-01
## X698     8.669976e-02
## X699     3.873514e-03
## X700     0.000000e+00
## X701     0.000000e+00
## X702     0.000000e+00
## X703     0.000000e+00
## X704     9.550031e-02
## X705     8.408505e-01
## X706     1.646284e+00
## X707     5.004919e+00
## X708     1.257717e+01
## X709     2.256066e+01
## X710     2.384051e+01
## X711     3.844464e+01
## X712     4.068766e+01
## X713     4.714183e+01
## X714     3.429529e+01
## X715     2.082394e+01
## X716     2.170802e+01
## X717     1.828884e+01
## X718     1.809338e+01
## X719     1.434132e+01
## X720     7.448370e+00
```

```
## X721     3.139993e+00
## X722     1.493530e+00
## X723     5.037456e-01
## X724     2.436979e-01
## X725     8.730474e-02
## X726     3.951325e-03
## X727     7.773791e-03
## X728     0.000000e+00
## X729     0.000000e+00
## X730     0.000000e+00
## X731     0.000000e+00
## X732     0.000000e+00
## X733     9.175018e-02
## X734     2.478611e-01
## X735     4.093424e-01
## X736     1.618869e+00
## X737     3.821462e+00
## X738     3.999173e+00
## X739     5.064057e+00
## X740     6.636494e+00
## X741     1.093186e+01
## X742     9.450321e+00
## X743     7.761628e+00
## X744     7.925880e+00
## X745     5.733391e+00
## X746     5.078652e+00
## X747     2.708270e+00
## X748     1.604313e+00
## X749     1.109834e+00
## X750     4.298241e-01
## X751     1.685948e-01
## X752     5.280699e-02
## X753     3.669606e-03
## X754     0.000000e+00
## X755     0.000000e+00
## X756     0.000000e+00
## X757     0.000000e+00
## X758     0.000000e+00
## X759     0.000000e+00
## X760     0.000000e+00
## X761     0.000000e+00
## X762     1.087821e-02
## X763     7.520844e-03
## X764     3.097568e-02
## X765     6.107249e-02
## X766     4.950666e-02
## X767     1.158761e-01
## X768     1.357511e-01
## X769     1.573506e-01
## X770     2.063532e-01
## X771     1.581600e-01
## X772     1.699874e-01
## X773     2.275615e-01
## X774     1.891100e-01
```

```
## X775     1.187392e-01
## X776     6.244529e-02
## X777     7.601105e-02
## X778     5.005349e-02
## X779     2.100194e-02
## X780     0.000000e+00
## X781     0.000000e+00
## X782     0.000000e+00
## X783     0.000000e+00
## X784     0.000000e+00
```

```
varImpPlot(modelo_rf)
```



A seguir, podemos ver OS cinco pixels mais importantes e as cinco menos importantes para a predição.

```
# importancia das variaveis
a <- as.data.frame(importance(modelo_rf))
head(a %>% arrange(desc(MeanDecreaseGini)),5)
```

```
##      MeanDecreaseGini
## X379         375.1617
## X351         356.4352
## X407         352.4710
## X410         323.9503
## X434         317.9197
```

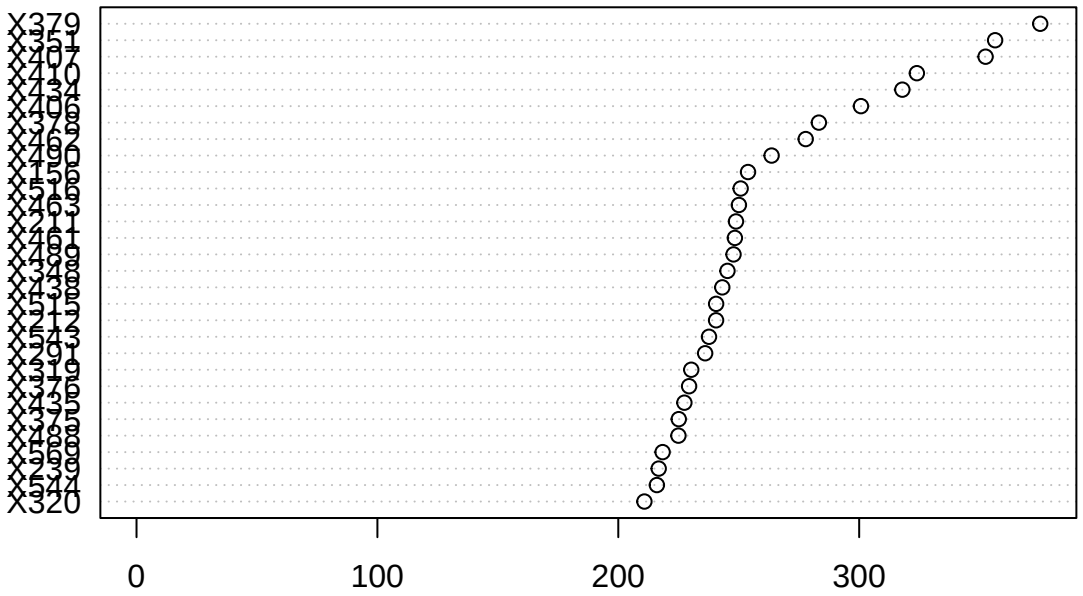
```
tail(a %>% arrange(desc(MeanDecreaseGini)),5)
```

```
##      MeanDecreaseGini
## X780                0
## X781                0
## X782                0
## X783                0
## X784                0
```

```
rm(a)
```

Ainda, é possível vizualizar melhor no gráfico a seguir a importância de cada pixel.

```
varImpPlot(modelo_rf)
```



Percebemos a existência de pixels cuja contribuição para a previsão é inexistente.

```
y_rf_dentro<- predict(modelo_rf, db_digitos[ids,-1],type="class")
y_rf_fora<- predict(modelo_rf, db_digitos[-ids,-1],type="class")
```

```
# acuracia
rf_ac_dentro <- mean(y_rf_dentro == db_digitos$y_treino[ids])
rf_ac_fora <- mean(y_rf_fora == db_digitos$y_treino[-ids])
```

```
# resultados <- data.frame("Floresta Aleatoria",rf_ac_dentro, rf_ac_fora)
# names(resultados) <- c("modelo", "acuracia_dentro", "acuracia_fora")

# atualizacao do dataframe de resultados
resultados <- rbind(resultados,data.frame(modelo="Floresta aleatoria",acuracia_dentro=rf_ac_dentro, acu
resultados <- resultados %>% mutate(across(where(is.numeric),round,4))
resultados
```

```
##                          modelo acuracia_dentro acuracia_fora
## 1          Lasso - Lambda minimo          0.9364        0.9219
## 2 Lasso - Lambda 1 desvio padrao          0.9342        0.9207
## 3             Floresta aleatoria          1.0000        0.9697
```

Redes neurais

Para o modelo de redes neurais, foi considerado um modelo com duas camadas ocultas com 256 e 128 unidades respectivamente com função de ativação ReLU.

Como temos mais de uma classificação, foi utilizada a função softmax como função de ativação na camada de saída.

Ainda, foi considerada supressão de 40%, 30 epochs e tamanho de batch 128 de forma a tentar otimizar o tempo de processamento do modelo.

```
# library(keras)

X <- model.matrix(db_digitos[,1] ~ .,
```

```
                  data = db_digitos[,-1]) # X deve ser uma matrix

# reshape
# library(Corbi)
# x_train <- submatrix(x_treino,ids,1:784)
# x_test <- submatrix(x_treino,-ids,1:784)

x_train <- X[ids,]
x_test <- X[-ids,]

# rescale
x_train <- x_train / 255
x_test <- x_test / 255
```

```
y_train <- to_categorical(y_treino[ids], 10)
```

```
## Loaded Tensorflow version 2.5.0
```

```
y_test <- to_categorical(y_treino[-ids], 10)
```

```
# Rede Neural ------------------------------------------------------------
# library(keras)

# primeiro passo: definir a estrutura que descreve a rede neural
# a ultima camada tem somente uma unidade

modelo_rn <- keras_model_sequential()
modelo_rn %>%
  layer_dense(units = 256, # numero de unidades
              activation = "relu", # funcao de ativacao
              input_shape = ncol(x_train)) %>% # dimensao da entrada
  layer_dropout(rate = 0.4) %>% # taxa de nos nos suprime em cada etapa
  layer_dense(units = 128, activation = 'relu') %>%
  layer_dropout(rate = 0.4) %>%
  layer_dense(units = 10, # camadas de saida com uma unica unidade pq eh um modelo de classificacao com
              activation = "softmax")

modelo_rn
```

```
# segundo passo: especificacoes que controlam o algoritmo de estimacao
modelo_rn %>%
  compile(loss = "categorical_crossentropy",   # funcao de custo
          optimizer = optimizer_rmsprop(),  # otimizador
          metrics = c('accuracy'))   # metrica para avaliar o erro
# a funcao compile() nao muda a variavel R, mas
# comunica as especificacoes para a instancia
# python correspondente que foi criada
```

```
# salvar o modelo
save_model_hdf5(modelo_rn, "mnist_rn")
```

```
# terceiro passo: ajustar o modelo (estimar os parametros)
history <- modelo_rn %>%
  fit(x_train, # preditoras de treino    dados de entrada
      y_train,  # resposta de treino
      batch_size = 128, # quantas observacoes escolhidas aleatoriamentes
```

```
                      # em cada passo do SGD
    epochs = 30, # uma epoca e' numero de passos do SGD 30
                    # para processar todos dados de treino
                    # nesse caso, cada epoca tem
                    # num de obs/batch passos pra completar uma epoca
    verbose=1,
#     callback = callback_early_stopping(monitor = "val_loss",
# min_delta = 0,
# patience = 20,
# verbose = 0,
# mode = c("auto"),
# baseline = NULL,
# restore_best_weights = FALSE),
    # dados de validacao para avaliar o progresso do modelo
    validation_data = list(x_test,
                           y_test))
```

```
saveRDS(history, "mnist_history.rds")
```

```
# carregar o modelo
modelo_rn <- load_model_hdf5("mnist_rn")
# history <- readRds("mnist_history.rds")
```

```
rn_ac_dentro <- modelo_rn %>% evaluate(x_train, y_train)
rn_ac_fora <- modelo_rn %>% evaluate(x_test, y_test)
```

```
# predicao

y_rn_dentro <- modelo_rn %>% predict(x_train) %>% k_argmax() %>% as.integer()
y_rn_fora <- modelo_rn %>% predict(x_test) %>% k_argmax() %>% as.integer()
```

```
# atualizacao do dataframe de resultados
resultados <- rbind(resultados,data.frame(modelo="Redes Neurais",acuracia_dentro=rn_ac_dentro[2], acura
resultados <- resultados %>% mutate(across(where(is.numeric),round,4))
resultados
```

```
##                                 modelo acuracia_dentro acuracia_fora
## 1                Lasso - Lambda minimo          0.9364        0.9219
## 2        Lasso - Lambda 1 desvio padrao          0.9342        0.9207
## 3                   Floresta aleatoria          1.0000        0.9697
## accuracy                 Redes Neurais          0.9971        0.9794
```

Percebemos o modelo de redes neurais foi o que obteve a maior acurácia tanto dentro como fora (conjunto de validação). Dessa forma, o modelo de redes neurais foi o modelo usado para realizar as previsões do conjunto de teste.

```
predicoes <- modelo_rn %>% predict(x_teste) %>% k_argmax() %>% as.integer
```

```
# Salva as predicoes em arquivo csv
write.csv(data.frame(y_pred = predicoes), file="mnist_9297784_yugooyama.csv")
```