Choose your own project

KC Cardakli

2024-08-07

Overview

In this project, we look at the US COVID deaths from Jan 2024 - Sept 23, 2023 and associated diagnoses at the time of death. We fit models using the associated diagnoses predicting the number of deaths. Once we find the best models we identify the variable importance for these models.
We attempt to identify the correlated factors in COVID19 deaths.

The input to our models are the number of diagnoses for each row and the output is the COVID19 deaths for that time period and geographic location.

We use dataset from US Health and Human Services (HHS). We first wrangle the data and eliminate duplicate data and rows with missing data. Please refer to the section "About the dataset" for further details.

We then convert the data into wide format and check against the total COVID19 deaths from CDC website. We construct 14 models and train these models via bootsrapping and cross validation with a k value of 25.

The models that provide best RMSE (root of mean square error) values are provided by the xgbTree and rf algorithms followed by gamLoess. xgbTree and rf are Tree-Based Models, xgbTree is Extreme Gradient Boosting Model where as rf is Random Forest Model. They provide similar variable importance. The third best model, gamLoess, combines Generalized Additive Models (GAM) with LOESS (Locally Estimated Scatterplot Smoothing) and gives a different variable importance. Please see below for details.

The neural network model(nnet), provided one the worst models with an RMSE value of 2186. We investigate to see if we can improve on this model by changing the model parameters, number of units in the hidden layer and decay. Although we improve the RMSE value significantly, it is still not as accurate as xgbTree, rf, or gamLoess models. Apparently nnet model can have only one hidden layer, and that may be limitation of this model.

When looking at the results of this project, It is probably important remember that correlation doesn't mean causality; some of the diagnoses could be (and are) complications of COVID19, rather than causes. The best model gives us the 10 most important variables, in the order of decreasing importance, as respiratory failure(COVID19 itself causes respiratory failure), influenza and pneumonia, malignant neoplasms (cancer), adult respiratory distress syndrome, cardiac arrhythmia, respiratory arrest, chronic lower respiratory diseases, diabetes, ischemic heart disease, and heart failure. The result seems reasonable, but of course much more research would have to be done to generalize.

It would be very interesting to get the list of the diagnoses when the patient was diagnosed with COVID19 not at the time of death. And of course, it would be even better if we could get the diagnoses of every COVID19 patient when they got sick, not just the ones that died from it.

About The Dataset

For this project we use a public dataset from US Health and Human Services (HHS):

Conditions Contributing to COVID-19 Deaths, by State and Age, Provisional 2020-2023 https://catalog.data.gov/dataset/conditions-contributing-to-deaths-involving-coronavirus-disease-2019-covid-19-by-age-group

The dataset summarizes the COVID19 deaths and associated factors.

I suspect the underlying full dataset is at the US Center for Disease Control (CDC):

https://wonder.cdc.gov/mcd.html

but this database is not public, and only available to researchers with certain conditions:

https://wonder.cdc.gov/mcd-icd10-provisional.html

Among the conditions is "Do not present or publish death counts of 9 or fewer or death rates based on counts of nine or fewer (in figures, graphs, maps, tables, etc.)."

I have asked the TAs of this course about the CDC dataset and I was asked not use CDC dataset since I could not provide the dataset as part of the project.

HHS dataset puts NA for values 1-9 as CDC dataset conditions dictate.

HHS dataset also provides the national sums.

HHS dataset provides the associated diagnoses for COVID19 deaths in each row, data is tabulated as on a per month / state basis.

In the State column, possible values are all US states, Washington DC, and Puerto Rico.

Methods

We use 14 models:	
lm	Linear Regression
glm	Generalized Linear Model
knn	K-Nearest Neighbors
rf	Random Forest
gamLoess	Generalized Additive Model (GAM) combined with LOESS (Locally Estimated Scatterplot Smoothing)
rpart	Recursive Partitioning and Regression Trees
xgbTree	Extreme Gradient Boosting (XGBoost)
cforest	Conditional Inference Trees
glmnet	Regularized Generalized Linear Models (Elastic Net)
bayesglm	Bayesian Generalized Linear Models
pcr	Principal Component Regression
pls	Partial Least Squares Regression
ridge	Ridge Regression
nnet	Neural Network

We first start with bootstrapping k value of 25, and then we use cross validation with a k-value of 25.

Results

Cross validation with a k value of 25 provides the best RMSE values on the test data.

Model	RMSE
xgbTree	161.1663

Model	RMSE
rf	248.5894
gamLoess	252.0772

Bootstrapping yields similar results; the order of models do not change:

Model	RMSE
xgbTree	231.0553
rf	248.5894
gamLoess	252.0772

xgbTree variable importance gives us the model's correlated diagnoses	
Respiratory.failure	100.00000
Influenza.pneumonia	90.07434
Malignant.neoplasms	30.88395
Adult.respiratory.distress.syndrome	21.18099
Cardiac.arrhythmia	18.13848
Respiratory.arrest	14.79050
Chronic.lower.respiratory.diseases	12.53784
Diabetes	5.39707
Ischemic.heart.disease	5.32439
Heart.failure	4.20999
Hypertensive.diseases	2.35590
Vascular.unspecified.dementia	1.06199
Cardiac.arrest	0.32535
Sepsis	0.29698
Other.diseases.of.the.respiratory.system	0.22408
Cerebrovascular.diseases	0.17067
Renal.failure	0.13228
Obesity	0.10758
Alzheimer.disease	0.03441
Other.diseases.of.the.circulatory.system	0.00000

Variable Importance for rf model is similar:	
Influenza.pneumonia	100.0000
Respiratory.failure	84.0869
Ischemic.heart.disease	67.4816
Cardiac.arrhythmia	44.0161
Other.diseases.of.the.circulatory.system	23.0288
Cerebrovascular.diseases	16.9564
Renal.failure	16.5231
Diabetes	13.6520
Adult.respiratory.distress.syndrome	13.1184
Respiratory.arrest	9.0165
Chronic.lower.respiratory.diseases	7.5493
Heart.failure	6.6621
Hypertensive.diseases	5.5820
Cardiac.arrest	4.9089

Variable Importance for rf model is similar:	
Alzheimer.disease	3.1151
Malignant.neoplasms	2.4882
Other.diseases.of.the.respiratory.system	2.2472
Sepsis	1.5487
Vascular.unspecified.dementia	0.9601
Obesity	0.0000

Variable Importance for gamLoess model is completely different:	
Other.diseases.of.the.circulatory.system	100.000
Cardiac.arrest	84.522
Renal.failure	83.760
Sepsis	80.355
Cerebrovascular.diseases	68.651
Chronic.lower.respiratory.diseases	57.188
Diabetes	51.617
Obesity	49.864
Adult.respiratory.distress.syndrome	49.663
Respiratory.arrest	45.461
Hypertensive.diseases	35.356
Respiratory.failure	34.682
Other.diseases.of.the.respiratory.system	27.160
Influenza.pneumonia	20.816
Alzheimer.disease	17.822
Cardiac.arrhythmia	16.083
Heart.failure	12.874
Malignant.neoplasms	8.244
Vascular.unspecified.dementia	3.550
Ischemic.heart.disease	0.000

Conclusions

Tree based models give us the best RMSE results. In particular xgbTree model gives the best results with 25-fold cross validation. Considering the mean of the output parameter, number of COVID19 deaths, is 1828.651, and median is 1236, a RMSE of 161.1663 seems good.
Of course, it may be possible to improve on this RMSE value by using the base source data from CDC.

It was also interesting to see that tree models have similar variable importance, while gamLoess has a much different variable importance

Possible Future Work

It would be interesting to replicate the work with the non-public base data from CDC with the actual counts 1-9 included. That would give us more data rows, and it would be more precise.

Also as mentioned before, it would be interesting to get the diagnoses of the patients when they get contract COVID19, then we can compare the possible associations (possibly risk factors) between the patients that survived COVID19 and not.

References

Public dataset from US Health and Human Services (HHS): Conditions Contributing to COVID-19 Deaths, by State and Age, Provisional 2020-2023

https://catalog.data.gov/dataset/conditions-contributing-to-deaths-involving-coronavirus-disease-2019-covid-19-by-age-group

Libraries

```
##########################################################################
#
# Please uncomment these lines to install packages as needed in your system:
#
#install.packages("dplyr")
#install.packages("data.table")
#install.packages("readr")
#install.packages("caret")
#install.packages("tidyr")

library("dplyr")
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library("data.table")
```

```
##
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
library("readr")
library("caret")
```

```
## Loading required package: ggplot2

## Loading required package: lattice
```

```
library("tidyr")

options(max.print = 500)
```

Reading the HHS dataset and checking one month and one state visually

Please don't forget to modify your working directory as needed. The R, Rmd, and pdf files and csv file from HHS should be in your working directory.

```
##########################################################################
# !!!!
# IMPORTANT: You may modify the directory to your working directory as needed
#
print("working directory: ")
```

```
## [1] "working directory: "
```

```
# setwd("/Users/kccardakli/Documents/R/projects/Capstone Project/CYO")
getwd()
```

```
## [1] "/Users/kccardakli/Documents/R/projects/Capstone Project/CYO"
```

```
options(timeout = 1200)

# Specifying the URL of the uncompressed file from CDC
# If this fails for you, you can download the zipped file from GitHub, and unzip it manually:
# https://github.com/KCardakli/CYO/blob/4c86003ad5095c7baceed361753182a35b30a439/Conditions_Contributin
# or the uncompressed file from Google drive:
# https://drive.google.com/file/d/1WQt3y2N-xiEfEiuI4VuExcexpjjchNe2/view?usp=sharing
# R-studio fails when large files are downloaded from GitHub or Google drive:
url <- "https://data.cdc.gov/api/views/hk9y-quqm/rows.csv?accessType=DOWNLOAD"

# Name of the downloaded file
destfile <- "Conditions_Contributing_to_COVID-19_Deaths__by_State_and_Age__Provisional_2020-2023.csv"

# Downloading the file
if(!file.exists(destfile))
   download.file(url, destfile, method = "libcurl")

# reading the HHS dataset
data <- read.csv("./Conditions_Contributing_to_COVID-19_Deaths__by_State_and_Age__Provisional_2020-2023

# Let's check the data for one month and one state
# We will use the Age.Group == "All Ages" as it minimizes the effect of putting NA for
# values 1-9 as explained above.
#
temp <- data |> filter(Month == 8 & Year == 2020 & State == "Alabama" & Age.Group == "All Ages") |>
  select(-Data.As.Of, -Start.Date, -End.Date, -ICD10_codes)
write_csv(temp, "temp.csv")
```

Data Wrangling

```r
# Eliminate the nationwide data as they replicate the statewide data by summation.
# We don't want to double count the data.
# We will use the monthly data.
# Eliminate the rows that have "" or NA for the COVID.19.Deaths, this comes from the CDC
#  source data where "one or more data cells have counts between 1-9 and have been suppressed
#  in accordance with NCHS confidentiality standards".  This elimination introduces some error,
#  solution would be to use the CDC dataset, but it is not public.
# We also eliminate  "All other conditions and causes (residual)", "Intentional and unintentional
#  injury, poisoning, and other adverse events" as these are catchall phrases and are not specific
#  diagnoses.
# We are using the COVID.19.Deaths and not Number.of.Mentions as some conditions are
#  mentioned multiple times in death certificates under reason for death and contributing
#  factors, so they could be counted more than once.
data <- data |> filter(data$Group == "By Month" &
      data$COVID.19.Deaths != "" &
      State != "United States" &
      !is.na(data$COVID.19.Deaths)  &
      Condition != 'All other conditions and causes (residual)'  &
      Condition != 'Intentional and unintentional injury, poisoning, and other adverse events'  &
      Age.Group == 'All Ages') |> select(Year, Month, State, COVID.19.Deaths, Condition)
str(data)
```

```
## 'data.frame':    35091 obs. of  5 variables:
##  $ Year           : int  2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 ...
##  $ Month          : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ State          : chr  "Alabama" "Alabama" "Alabama" "Alabama" ...
##  $ COVID.19.Deaths: int  0 0 11 92 128 125 264 318 188 183 ...
##  $ Condition      : chr  "Influenza and pneumonia" "Influenza and pneumonia" "Influenza and pneumoni
```

```r
# Let's check the total number of COVID deaths in the dataset
# It is inline with other resources from CDC (1.14M):
# https://www.cdc.gov/nchs/nvss/vsrr/covid19/index.htm
#
t <- data |> filter(Condition == "COVID-19" & !is.na(COVID.19.Deaths))
print("Total COVID deaths in the dataset  01/01/2020 - 09/23/2023:")
```

```
## [1] "Total COVID deaths in the dataset  01/01/2020 - 09/23/2023:"
```

```r
sum(t$COVID.19.Deaths)
```

```
## [1] 1152167
```

```r
# We now rename the fields that will become columns so that they are easily readable and can be
#   accessed without using `` characters
data <- data |>
  mutate(Condition = gsub(",", "", Condition)) |>
  mutate(Condition = gsub(" and ", " ", Condition)) |>
  mutate(Condition = gsub('-', "", Condition)) |>
  mutate(Condition = gsub("  ", " ", Condition)) |>
  mutate(Condition = gsub(" ", ".", Condition))

#These are the unique conditions
unique(data$Condition)
```

```
##  [1] "Influenza.pneumonia"
##  [2] "Chronic.lower.respiratory.diseases"
##  [3] "Adult.respiratory.distress.syndrome"
##  [4] "Respiratory.failure"
##  [5] "Respiratory.arrest"
##  [6] "Other.diseases.of.the.respiratory.system"
##  [7] "Hypertensive.diseases"
##  [8] "Ischemic.heart.disease"
##  [9] "Cardiac.arrest"
## [10] "Cardiac.arrhythmia"
## [11] "Heart.failure"
## [12] "Cerebrovascular.diseases"
## [13] "Other.diseases.of.the.circulatory.system"
## [14] "Sepsis"
## [15] "Malignant.neoplasms"
## [16] "Diabetes"
## [17] "Obesity"
## [18] "Alzheimer.disease"
## [19] "Vascular.unspecified.dementia"
## [20] "Renal.failure"
## [21] "COVID19"
```

```r
# Form the wide data with "Condition"s are columns
wide_data <- data |>
  pivot_wider(id_cols = c(Year, Month, State),
              names_from = Condition,
              values_from = c(COVID.19.Deaths))
str(wide_data)
```

```
## tibble [2,385 x 24] (S3: tbl_df/tbl/data.frame)
##  $ Year                                    : int [1:2385] 2020 2020 2020 2020 2020 2020 2020 2020 20
##  $ Month                                   : int [1:2385] 1 2 3 4 5 6 7 8 9 10 ...
##  $ State                                   : chr [1:2385] "Alabama" "Alabama" "Alabama" "Alabama" ..
##  $ Influenza.pneumonia                     : int [1:2385] 0 0 11 92 128 125 264 318 188 183 ...
##  $ Chronic.lower.respiratory.diseases      : int [1:2385] 0 NA NA 36 32 31 73 55 26 30 ...
##  $ Adult.respiratory.distress.syndrome     : int [1:2385] 0 0 NA 32 24 28 42 67 38 40 ...
##  $ Respiratory.failure                     : int [1:2385] 0 0 21 126 144 134 299 380 234 245 ...
##  $ Respiratory.arrest                      : int [1:2385] NA 0 0 14 39 32 74 71 37 39 ...
##  $ Other.diseases.of.the.respiratory.system: int [1:2385] 0 0 NA 15 NA 17 26 34 22 22 ...
##  $ Hypertensive.diseases                   : int [1:2385] NA 0 NA 52 51 35 117 105 55 51 ...
##  $ Ischemic.heart.disease                  : int [1:2385] 0 0 NA 31 39 32 82 82 44 49 ...
##  $ Cardiac.arrest                          : int [1:2385] NA 0 NA 43 60 68 116 128 61 68 ...
##  $ Cardiac.arrhythmia                      : int [1:2385] NA 0 NA 24 27 18 60 58 31 33 ...
##  $ Heart.failure                           : int [1:2385] 0 0 NA 23 36 16 58 55 35 38 ...
##  $ Cerebrovascular.diseases                : int [1:2385] 0 0 NA 20 22 23 30 41 21 27 ...
##  $ Other.diseases.of.the.circulatory.system: int [1:2385] NA NA NA 18 27 15 42 61 40 35 ...
##  $ Sepsis                                  : int [1:2385] 0 0 10 51 54 51 87 112 70 68 ...
##  $ Malignant.neoplasms                     : int [1:2385] 0 0 NA 11 12 12 35 12 24 22 ...
##  $ Diabetes                                : int [1:2385] NA NA NA 39 47 42 95 110 68 58 ...
##  $ Obesity                                 : int [1:2385] 0 0 0 NA NA NA 26 30 16 10 ...
##  $ Alzheimer.disease                       : int [1:2385] 0 0 NA 15 18 NA 44 31 16 17 ...
##  $ Vascular.unspecified.dementia           : int [1:2385] 0 0 NA 31 50 35 68 71 65 39 ...
##  $ Renal.failure                           : int [1:2385] 0 0 NA 28 47 38 96 82 57 48 ...
##  $ COVID19                                 : int [1:2385] NA NA 48 339 430 399 879 964 568 560 ...
```

```r
wide_data |> print(, n = 25)
```

```
## # A tibble: 2,385 x 24
##     Year Month State   Influenza.pneumonia Chronic.lower.respiratory.diseases
##    <int> <int> <chr>                 <int>                             <int>
##  1  2020     1 Alabama                   0                                 0
##  2  2020     2 Alabama                   0                                NA
##  3  2020     3 Alabama                  11                                NA
##  4  2020     4 Alabama                  92                                36
##  5  2020     5 Alabama                 128                                32
##  6  2020     6 Alabama                 125                                31
##  7  2020     7 Alabama                 264                                73
##  8  2020     8 Alabama                 318                                55
##  9  2020     9 Alabama                 188                                26
## 10  2020    10 Alabama                 183                                30
## 11  2020    11 Alabama                 217                                51
## 12  2020    12 Alabama                 546                               153
## 13  2021     1 Alabama                 838                               175
## 14  2021     2 Alabama                 334                                65
## 15  2021     3 Alabama                 138                                32
## 16  2021     4 Alabama                  70                                15
## 17  2021     5 Alabama                  74                                18
## 18  2021     6 Alabama                  43                                NA
## 19  2021     7 Alabama                  93                                16
## 20  2021     8 Alabama                 619                                81
## 21  2021     9 Alabama                 740                               113
## 22  2021    10 Alabama                 316                                58
## 23  2021    11 Alabama                 100                                19
## 24  2021    12 Alabama                  93                                27
## 25  2022     1 Alabama                 367                               126
## # i 2,360 more rows
## # i 19 more variables: Adult.respiratory.distress.syndrome <int>,
## #   Respiratory.failure <int>, Respiratory.arrest <int>,
## #   Other.diseases.of.the.respiratory.system <int>,
## #   Hypertensive.diseases <int>, Ischemic.heart.disease <int>,
## #   Cardiac.arrest <int>, Cardiac.arrhythmia <int>, Heart.failure <int>,
## #   Cerebrovascular.diseases <int>, ...
```

```r
write_csv(wide_data, "wide_data_export.csv")

# We have lots of values with NA values
# We could assume some values, but that would introduce significant errors to the models.
# Instead, let's delete the rows that have NA values, this will reduce the rows in our dataset
# but there will be no errors introduced by the NA value due to counts of 1-9.
wide_data <- wide_data[(complete.cases(wide_data) == TRUE), ]
```

```r
# Now lets also delete the rows that have 0 COVID-19 deaths.
# This was common at the beginning of 2020.
wide_data <- wide_data |> filter(`COVID19` > 0)

# Change the Month column to be a linear parameter as it gives us a better parameter for the models,
#   no need for the Year parameter afterwards
# Experimented with using the month as a parameter as well
#  But it did not provide much improvement, so eliminated the Month parameter later on..
wide_data <- wide_data |> mutate(Month = (Year - 2020)*12 + Month) |> select(-Year, -State, -Month)

#This is our final dataset
str(wide_data)
```

```
## tibble [430 x 21] (S3: tbl_df/tbl/data.frame)
##  $ Influenza.pneumonia                     : int [1:430] 264 318 188 183 217 546 838 334 619 740 ...
##  $ Chronic.lower.respiratory.diseases      : int [1:430] 73 55 26 30 51 153 175 65 81 113 ...
##  $ Adult.respiratory.distress.syndrome     : int [1:430] 42 67 38 40 51 61 120 60 73 141 ...
##  $ Respiratory.failure                     : int [1:430] 299 380 234 245 321 694 962 404 717 861 ...
##  $ Respiratory.arrest                      : int [1:430] 74 71 37 39 62 158 196 86 175 193 ...
##  $ Other.diseases.of.the.respiratory.system: int [1:430] 26 34 22 22 26 71 90 54 51 75 ...
##  $ Hypertensive.diseases                   : int [1:430] 117 105 55 51 60 170 228 108 145 179 ...
##  $ Ischemic.heart.disease                  : int [1:430] 82 82 44 49 66 152 194 73 100 122 ...
##  $ Cardiac.arrest                          : int [1:430] 116 128 61 68 115 287 370 171 298 348 ...
##  $ Cardiac.arrhythmia                      : int [1:430] 60 58 31 33 64 122 181 59 106 117 ...
##  $ Heart.failure                           : int [1:430] 58 55 35 38 50 128 165 59 78 95 ...
##  $ Cerebrovascular.diseases                : int [1:430] 30 41 21 27 27 72 106 45 53 58 ...
##  $ Other.diseases.of.the.circulatory.system: int [1:430] 42 61 40 35 51 108 181 66 101 132 ...
##  $ Sepsis                                  : int [1:430] 87 112 70 68 89 174 284 139 153 240 ...
##  $ Malignant.neoplasms                     : int [1:430] 35 12 24 22 25 66 90 39 49 42 ...
##  $ Diabetes                                : int [1:430] 95 110 68 58 79 147 251 111 153 187 ...
##  $ Obesity                                 : int [1:430] 26 30 16 10 13 48 61 19 71 81 ...
##  $ Alzheimer.disease                       : int [1:430] 44 31 16 17 14 40 41 18 13 16 ...
##  $ Vascular.unspecified.dementia           : int [1:430] 68 71 65 39 38 94 108 42 30 33 ...
##  $ Renal.failure                           : int [1:430] 96 82 57 48 92 176 281 110 167 216 ...
##  $ COVID19                                 : int [1:430] 879 964 568 560 768 1749 2415 970 1682 1978
```

```r
write_csv(wide_data, "wide_data_export.csv")

print('Mean of our output Parameter, COVID19:')
```

```
## [1] "Mean of our output Parameter, COVID19:"
```

```r
mean(wide_data$COVID19)
```

```
## [1] 1828.651
```

```r
print('Median of our output Parameter, COVID19:')
```

```
## [1] "Median of our output Parameter, COVID19:"
```

```
median(wide_data$COVID19)
```

```
## [1] 1236
```

Preparing training and test datasets

Let's prepare our training and test datasets

Final hold-out test set will be 10% of the data

We use the naming convention as in the previous project

```
# Set the seed for reproducibility
set.seed(1, sample.kind="Rounding") # if using R 3.6 or later
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

```
# set.seed(1) # if using R 3.5 or earlier

test_index <- createDataPartition(y = wide_data$COVID19, times = 1, p = 0.1, list = FALSE)
edx <- wide_data[-test_index,]
final_holdout_test <- wide_data[test_index,]
```

Comparing the Models via Bootstrapping with a k value of 25

```
##########################################################################
#
# Comparing models with bootstrapping method with a value of 25
# On my computer this subsection takes 2 minutes 50 seconds
#

# Define the trainControl with bootstrapping
train_control <- trainControl(
  method = "boot",      # Bootstrapping method
  number = 25,          # Number of bootstrap samples
  verboseIter = FALSE   # Set to TRUE to see the progress
)

# List of different model types to run
models_list <- c("lm", "glm", "knn", "rf", "gamLoess", "rpart", "xgbTree",
                 "cforest", "glmnet", "bayesglm", "pcr", "pls", "ridge", "nnet")

# Create empty lists to store the results
boot_model_results <- list()
boot_rmse <- list()

# Loop over the list of models and train each one
# suppress the warnings about deprecated functions
```

```r
# and nnet training messsages
for (model_type in models_list) invisible(capture.output({
  # to be able to replicate the results
  set.seed(1, sample.kind="Rounding") # if using R 3.6 or later
  # set.seed(1) # if using R 3.5 or earlier

  model <- train(
    COVID19 ~ .,               # COVID19 variable vs all predictors
    data = edx,                # Training dataset
    method = model_type,       # Current model from the list
    trControl = train_control # Use the defined trainControl
  )

  # Predict on the final_holdout_test dataset
  predictions <- predict(model, newdata = final_holdout_test)

  boot_rmse[[model_type]] <- sqrt(mean((predictions - final_holdout_test$COVID19)^2))

  # Store the result in the list
  boot_model_results[[model_type]] <- model
}))
```

```
## Loading required package: gam

## Loading required package: splines

## Loading required package: foreach

## Loaded gam 1.22-3
```

```r
# Access results of each model
boot_rmse
```

```
## $lm
## [1] 291.7629
##
## $glm
## [1] 291.7629
##
## $knn
## [1] 252.6973
##
## $rf
## [1] 248.5894
##
## $gamLoess
## [1] 252.0772
##
## $rpart
## [1] 888.8269
##
## $xgbTree
## [1] 231.0553
##
## $cforest
## [1] 263.3392
##
## $glmnet
## [1] 304.0771
##
## $bayesglm
## [1] 291.7629
##
## $pcr
## [1] 342.2447
##
## $pls
## [1] 294.813
##
## $ridge
## [1] 291.7629
##
## $nnet
## [1] 2186.664
```

```
# Let's look at the best models and worst models

boot_top_3_rmse <- sort(unlist(boot_rmse), decreasing = FALSE)[1:3]
boot_top_3_rmse
```

```
##  xgbTree       rf gamLoess
## 231.0553 248.5894 252.0772
```

```
boot_worst_3_rmse <- sort(unlist(boot_rmse), decreasing = TRUE)[1:3]
boot_worst_3_rmse
```

```
##      nnet     rpart       pcr
## 2186.6641  888.8269  342.2447
```

```
# Extract the variable importance for top 3 models
for (model_name in names(boot_top_3_rmse)) {
  cat("Variable Importance for model:", model_name, "\n")
  var_importance <- varImp(boot_model_results[[model_name]])
  print(var_importance)
  cat("\n")
}
```

```
## Variable Importance for model: xgbTree
## xgbTree variable importance
##
##                                           Overall
## Influenza.pneumonia                     100.00000
## Renal.failure                            68.90390
## Respiratory.failure                      45.86465
## Adult.respiratory.distress.syndrome       26.08815
## Respiratory.arrest                        16.18996
## Cardiac.arrhythmia                        14.16910
## Chronic.lower.respiratory.diseases         4.41823
## Ischemic.heart.disease                     4.28138
## Hypertensive.diseases                      2.35341
## Diabetes                                   2.32260
## Heart.failure                              1.18459
## Vascular.unspecified.dementia              0.77940
## Obesity                                    0.53307
## Cardiac.arrest                             0.38919
## Malignant.neoplasms                        0.38809
## Sepsis                                     0.22384
## Other.diseases.of.the.respiratory.system   0.16147
## Cerebrovascular.diseases                   0.03492
## Other.diseases.of.the.circulatory.system   0.02575
## Alzheimer.disease                          0.00000
##
## Variable Importance for model: rf
## rf variable importance
##
##                                           Overall
## Influenza.pneumonia                      100.0000
## Respiratory.failure                       84.0869
## Ischemic.heart.disease                    67.4816
## Cardiac.arrhythmia                        44.0161
## Other.diseases.of.the.circulatory.system  23.0288
## Cerebrovascular.diseases                  16.9564
## Renal.failure                             16.5231
## Diabetes                                  13.6520
## Adult.respiratory.distress.syndrome       13.1184
## Respiratory.arrest                         9.0165
## Chronic.lower.respiratory.diseases         7.5493
## Heart.failure                              6.6621
## Hypertensive.diseases                      5.5820
## Cardiac.arrest                             4.9089
## Alzheimer.disease                          3.1151
## Malignant.neoplasms                        2.4882
## Other.diseases.of.the.respiratory.system   2.2472
## Sepsis                                     1.5487
## Vascular.unspecified.dementia              0.9601
## Obesity                                    0.0000
##
## Variable Importance for model: gamLoess
## gamLoess variable importance
##
##                                          Overall
## Other.diseases.of.the.circulatory.system 100.000
## Cardiac.arrest                            84.522
## Renal.failure                             83.760
## Sepsis                                    80.355
## Cerebrovascular.diseases                  68.651
## Chronic.lower.respiratory.diseases        57.188
## Diabetes                                  51.617
## Obesity                                 49.864
## Adult.respiratory.distress.syndrome     49.663
## Respiratory.arrest                      45.461
## Hypertensive.diseases                   35.356
## Respiratory.failure                     34.682
## Other.diseases.of.the.respiratory.system  27.160
## Influenza.pneumonia                     20.816
## Alzheimer.disease                       17.822
## Cardiac.arrhythmia                      16.083
## Heart.failure                           12.874
## Malignant.neoplasms                      8.244
## Vascular.unspecified.dementia            3.550
## Ischemic.heart.disease                   0.000
```

Comparing the Models via 25-fold Cross-Validation

```r
# Define the trainControl with cross validation method
train_control <- trainControl(
  method = "cv",        # cross validation method
  number = 25,          # Number of bootstrap samples
  verboseIter = FALSE   # Set to TRUE to see the progress
)

# Create empty lists to store the results
cv_model_results <- list()
cv_rmse <- list()

# Loop over the list of models and train each one
# suppress the warnings about deprecated functions
# and nnet training messsages
for (model_type in models_list) invisible(capture.output({
  # to be able to replicate the results
  set.seed(1, sample.kind="Rounding") # if using R 3.6 or later
  # set.seed(1) # if using R 3.5 or earlier
  model <- train(
    COVID19 ~ .,                # COVID19 variable vs all predictors
    data = edx,                 # Training dataset
    method = model_type,        # Current model from the list
    trControl = train_control # Use the defined trainControl
  )

  # Predict on the final_holdout_test dataset
  predictions <- predict(model, newdata = final_holdout_test)

  cv_rmse[[model_type]] <- sqrt(mean((predictions - final_holdout_test$COVID19)^2))

  # Store the result in the list
  cv_model_results[[model_type]] <- model

}))
```

```r
# Access results of each model:
cv_rmse
```

```
## $lm
## [1] 291.7629
##
## $glm
## [1] 291.7629
##
## $knn
## [1] 252.6973
##
## $rf
## [1] 248.5894
##
## $gamLoess
## [1] 252.0772
##
## $rpart
## [1] 888.8269
##
## $xgbTree
## [1] 161.1663
##
## $cforest
## [1] 263.3392
##
## $glmnet
## [1] 304.0771
##
## $bayesglm
## [1] 291.7629
##
## $pcr
## [1] 342.2447
##
## $pls
## [1] 294.813
##
## $ridge
## [1] 292.1945
##
## $nnet
## [1] 2186.664
```

```r
# Let's look at the best models and worst models
cv_top_3_rmse <- sort(unlist(cv_rmse), decreasing = FALSE)[1:3]
cv_top_3_rmse
```

```
##  xgbTree       rf gamLoess
## 161.1663 248.5894 252.0772
```

```r
cv_worst_3_rmse <- sort(unlist(cv_rmse), decreasing = TRUE)[1:3]
cv_worst_3_rmse
```

```
##      nnet     rpart       pcr
## 2186.6641  888.8269  342.2447
```

Extract the variable importance for top 3 models for 25-fold cross validation

```r
# Extract the variable importance for top 3 models
for (model_name in names(cv_top_3_rmse)) {
  cat("Variable Importance for model:", model_name, "\n")
  var_importance <- varImp(cv_model_results[[model_name]])
  print(var_importance)
  cat("\n")
}
```

```
## Variable Importance for model: xgbTree
## xgbTree variable importance
##
##                                          Overall
## Respiratory.failure                    100.00000
## Influenza.pneumonia                     90.07434
## Malignant.neoplasms                     30.88395
## Adult.respiratory.distress.syndrome     21.18099
## Cardiac.arrhythmia                      18.13848
## Respiratory.arrest                      14.79050
## Chronic.lower.respiratory.diseases      12.53784
## Diabetes                                 5.39707
## Ischemic.heart.disease                   5.32439
## Heart.failure                            4.20999
## Hypertensive.diseases                    2.35590
## Vascular.unspecified.dementia            1.06199
## Cardiac.arrest                           0.32535
## Sepsis                                   0.29698
## Other.diseases.of.the.respiratory.system 0.22408
## Cerebrovascular.diseases                 0.17067
## Renal.failure                            0.13228
## Obesity                                  0.10758
## Alzheimer.disease                        0.03441
## Other.diseases.of.the.circulatory.system 0.00000
##
## Variable Importance for model: rf
## rf variable importance
##
##                                         Overall
## Influenza.pneumonia                    100.0000
## Respiratory.failure                     84.0869
## Ischemic.heart.disease                  67.4816
## Cardiac.arrhythmia                      44.0161
## Other.diseases.of.the.circulatory.system  23.0288
## Cerebrovascular.diseases                   16.9564
## Renal.failure                              16.5231
## Diabetes                                   13.6520
## Adult.respiratory.distress.syndrome        13.1184
## Respiratory.arrest                          9.0165
## Chronic.lower.respiratory.diseases          7.5493
## Heart.failure                               6.6621
## Hypertensive.diseases                       5.5820
## Cardiac.arrest                              4.9089
## Alzheimer.disease                           3.1151
## Malignant.neoplasms                         2.4882
## Other.diseases.of.the.respiratory.system    2.2472
## Sepsis                                      1.5487
## Vascular.unspecified.dementia               0.9601
## Obesity                                     0.0000
##
## Variable Importance for model: gamLoess
## gamLoess variable importance
##
##                                           Overall
## Other.diseases.of.the.circulatory.system 100.000
## Cardiac.arrest                            84.522
## Renal.failure                             83.760
## Sepsis                                    80.355
## Cerebrovascular.diseases                  68.651
## Chronic.lower.respiratory.diseases        57.188
## Diabetes                                  51.617
## Obesity                                   49.864
## Adult.respiratory.distress.syndrome       49.663
## Respiratory.arrest                        45.461
## Hypertensive.diseases                     35.356
## Respiratory.failure                       34.682
## Other.diseases.of.the.respiratory.system  27.160
## Influenza.pneumonia                       20.816
## Alzheimer.disease                         17.822
## Cardiac.arrhythmia                        16.083
## Heart.failure                             12.874
## Malignant.neoplasms                        8.244
## Vascular.unspecified.dementia              3.550
## Ischemic.heart.disease                     0.000
```

The best models each way of training are

xgbTree,rf,gamLoess

The worst one is nnet with rmse value of 2186.6641

Is there a way to optimize the nnet model?

In this section, we change the number of units in the hidden layer

and delay parameters

```
###########################################################################
# Is there a way to optimize the nnet model?
# In this section, we change the number of units in the hidden layer
# and delay parameters
# This section takes 2 minutes 55 seconds on my computer.

# Define the control function with cross-validation
train_control <- trainControl(method = "cv",
                              number = 10,
                              verboseIter = FALSE)

# Define a grid of parameters to tune
tune_grid <- expand.grid(
  size = c(3, 4, 5, 6, 7, 8),     # Number of units in the hidden layer
  decay = c(0.08, 0.09, 0.1, 0.11, 0.12, 0.13, 0.14, 0.15, 0.16)
  # Regularization parameter to avoid overfitting
)
set.seed(1, sample.kind="Rounding") # if using R 3.6 or later
# set.seed(1) # if using R 3.5 or earlier

# Train the model using nnet and parameter tuning
model <- train(
  COVID19 ~ .,               # Formula: COVID19 variable vs all predictors
  data = edx,
  method = "nnet",
  trControl = train_control,
  tuneGrid = tune_grid,
  linout = TRUE,         # For regression, use linout = TRUE
  trace = FALSE,         # Avoid printing during training
  maxit = 1000           # Maximum number of iterations
)

# View the best model
print(model)
```

```
## Neural Network
##
## 386 samples
##  20 predictor
```

```
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 348, 347, 347, 347, 347, 347, ...
## Resampling results across tuning parameters:
##
##   size  decay  RMSE       Rsquared   MAE
##   3     0.08   1461.7288  0.4029400   900.9504
##   3     0.09   1304.5786  0.5561979   774.4989
##   3     0.10    892.4448  0.7081071   510.8347
##   3     0.11   1749.0244  0.4506128   973.2203
##   3     0.12   1420.1433  0.3894717   876.7805
##   3     0.13   1655.0017  0.4311037   941.2270
##   3     0.14   1104.6506  0.6282932   688.4912
##   3     0.15   1426.2666  0.4839861   857.9145
##   3     0.16   1451.3846  0.4050624   889.3228
##   4     0.08    970.6093  0.7991485   560.9270
##   4     0.09   1531.8160  0.4958368   932.6634
##   4     0.10   1549.9992  0.3944219   935.1142
##   4     0.11   1671.6249  0.3536426   949.9917
##   4     0.12   1129.2312  0.5827576   656.1703
##   4     0.13    991.0900  0.6130996   596.3633
##   4     0.14    867.6216  0.7692948   448.5631
##   4     0.15   1199.3275  0.5322934   758.9058
##   4     0.16   1297.4818  0.6228802   712.9032
##   5     0.08   1572.9826  0.4886247  1006.1169
##   5     0.09   1108.3497  0.7043281   642.1245
##   5     0.10   1461.2434  0.4473150   886.1084
##   5     0.11   1205.9990  0.5717576   764.2888
##   5     0.12   1311.1847  0.4309102   821.2130
##   5     0.13   1280.5665  0.5567895   697.3412
##   5     0.14   1158.1199  0.6538318   598.7541
##   5     0.15   1235.6017  0.5392565   755.1650
##   5     0.16   1621.5276  0.4033298   957.9995
##   6     0.08    961.2195  0.6477998   546.1922
##   6     0.09   1522.5762  0.4220780   944.0241
##   6     0.10   1665.0195  0.2887197  1005.9812
##   6     0.11    906.0025  0.6668108   579.8203
##   6     0.12   1350.3861  0.5198642   815.3678
##   6     0.13   1686.5648  0.2760743  1041.6823
##   6     0.14   1475.9919  0.4750050   859.4721
##   6     0.15   1234.6931  0.5149940   778.6927
##   6     0.16   1213.1139  0.6049648   653.6328
##   7     0.08   1455.2769  0.5317280   912.7069
##   7     0.09   1193.2547  0.5870303   769.2171
##   7     0.10   1122.7525  0.7330771   652.1454
##   7     0.11   1108.1631  0.6120605   669.3107
##   7     0.12   1554.5952  0.3402095   914.5665
##   7     0.13   1093.6465  0.6922732   636.5650
##   7     0.14   1298.6059  0.4531386   799.2176
##   7     0.15   1529.2012  0.4159417   949.1994
##   7     0.16   1149.9133  0.5122837   723.5033
##   8     0.08   1569.5018  0.4282105   960.9842
##   8     0.09   1352.8489  0.5861313   810.4373
##   8     0.10   1435.1006  0.4311863   874.9393
##   8     0.11   1556.8649  0.4417696   959.1880
##   8     0.12   1492.2099  0.3917677   959.4196
##   8     0.13   1287.9239  0.5348019   813.5884
##   8     0.14   1543.2916  0.4665072   860.6616
##   8     0.15   1351.4736  0.4433945   857.8633
##   8     0.16   1086.0209  0.5654945   658.5192
##
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were size = 4 and decay = 0.14.
```

```
print(model$bestTune)
```

```
##    size decay
## 16    4  0.14
```

```
model
```

```
## Neural Network
##
## 386 samples
##  20 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 348, 347, 347, 347, 347, 347, ...
## Resampling results across tuning parameters:
##
##   size  decay  RMSE       Rsquared   MAE
##   3     0.08   1461.7288  0.4029400   900.9504
##   3     0.09   1304.5786  0.5561979   774.4989
##   3     0.10    892.4448  0.7081071   510.8347
##   3     0.11   1749.0244  0.4506128   973.2203
##   3     0.12   1420.1433  0.3894717   876.7805
##   3     0.13   1655.0017  0.4311037   941.2270
##   3     0.14   1104.6506  0.6282932   688.4912
##   3     0.15   1426.2666  0.4839861   857.9145
##   3     0.16   1451.3846  0.4050624   889.3228
##   4     0.08    970.6093  0.7991485   560.9270
##   4     0.09   1531.8160  0.4958368   932.6634
##   4     0.10   1549.9992  0.3944219   935.1142
##   4     0.11   1671.6249  0.3536426   949.9917
##   4     0.12   1129.2312  0.5827576   656.1703
##   4     0.13    991.0900  0.6130996   596.3633
##   4     0.14    867.6216  0.7692948   448.5631
##   4     0.15   1199.3275  0.5322934   758.9058
##   4     0.16   1297.4818  0.6228802   712.9032
##   5     0.08   1572.9826  0.4886247  1006.1169
##   5     0.09   1108.3497  0.7043281   642.1245
##   5     0.10   1461.2434  0.4473150   886.1084
##   5     0.11   1205.9990  0.5717576   764.2888
##   5     0.12   1311.1847  0.4309102   821.2130
##   5     0.13   1280.5665  0.5567895   697.3412
##   5     0.14   1158.1199  0.6538318   598.7541
##   5     0.15   1235.6017  0.5392565   755.1650
##   5     0.16   1621.5276  0.4033298   957.9995
##   6     0.08    961.2195  0.6477998   546.1922
##   6     0.09   1522.5762  0.4220780   944.0241
##   6     0.10   1665.0195  0.2887197  1005.9812
##   6     0.11    906.0025  0.6668108   579.8203
##   6     0.12   1350.3861  0.5198642   815.3678
##   6     0.13   1686.5648  0.2760743  1041.6823
##   6     0.14   1475.9919  0.4750050   859.4721
##   6     0.15   1234.6931  0.5149940   778.6927
##   6     0.16   1213.1139  0.6049648   653.6328
##   7     0.08   1455.2769  0.5317280   912.7069
##   7     0.09   1193.2547  0.5870303   769.2171
##   7     0.10   1122.7525  0.7330771   652.1454
##   7     0.11   1108.1631  0.6120605   669.3107
##   7     0.12   1554.5952  0.3402095   914.5665
##   7     0.13   1093.6465  0.6922732   636.5650
##   7     0.14   1298.6059  0.4531386   799.2176
##   7     0.15   1529.2012  0.4159417   949.1994
##   7     0.16   1149.9133  0.5122837   723.5033
##   8     0.08   1569.5018  0.4282105   960.9842
##   8     0.09   1352.8489  0.5861313   810.4373
##   8     0.10   1435.1006  0.4311863   874.9393
##   8     0.11   1556.8649  0.4417696   959.1880
##   8     0.12   1492.2099  0.3917677   959.4196
##   8     0.13   1287.9239  0.5348019   813.5884
##   8     0.14   1543.2916  0.4665072   860.6616
##   8     0.15   1351.4736  0.4433945   857.8633
##   8     0.16   1086.0209  0.5654945   658.5192
##
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were size = 4 and decay = 0.14.
```

```r
# Predict on the final_holdout_test dataset
predictions <- predict(model$finalModel, newdata = final_holdout_test)

# improved RMSE:
sqrt(mean((predictions - final_holdout_test$COVID19)^2))
```

```
## [1] 1492.863
```

```r
# RMSE has improved significantly, 1492.863
# but still it is much worse than our best model xgbTree: 161
# apparently nnet can only have one layer of hidden layer.
# To have multiple hidden layers, keras package from Google can be used apparently.
```