Project

Group

4/9/2020

```r
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
```

```r
library(readxl)
library(tidyr)
```

```r
# Read base data into R
# Cumulative number of confirmed cases and number of deaths
# Up until 4/25/2020
cv19.cty = read.csv("us-counties.csv")

cv19.st = cv19.cty %>%
  filter(date == "2020-04-25") %>%
  select(state, cases, deaths) %>%
  group_by(state) %>%
  summarise_all(sum) %>%
  filter(state != "Guam" &
           state != "Northern Mariana Islands" &
           state != "Puerto Rico" &
           state != "Virgin Islands") %>%
  mutate(PercentDeath = round((deaths/cases)*100, digits = 2)) %>%
  rename(State = state,
         Cases = cases,
         Deaths = deaths)

###

# Read hospital data into R
# Number of hospital beds (last updated 2020)
hospital.cty = read.csv("Hospitals.csv")

hospital.st = hospital.cty %>%
  filter(STATUS == "OPEN" & BEDS != -999) %>%
  select(STATE, BEDS) %>%
  group_by(STATE) %>%
  summarise_all(sum) %>%
  filter(STATE != "GU" &
           STATE != "MP" &
           STATE != "PR" &
```

```r
           STATE != "PW" &
           STATE != "VI") %>%
  mutate(STATE = state.name[match(STATE, state.abb)]) %>%
  mutate(STATE = replace_na(STATE, "District of Columbia")) %>%
  rename(State = STATE,
         NumBeds = BEDS)

###

# Read occupational data into R
# Number of registered nurses (2019)
occ = read_xlsx("state_M2019_dl.xlsx")

# Search for all titles related to nursing
# occ$occ_title[grep("Nurs", occ$occ_title)]

# Filter for registered nurse
nurse.st = occ %>%
  filter(occ_title == "Registered Nurses") %>%
  select(area_title, tot_emp) %>%
  slice(1:51) %>%
  mutate(RegNurse = as.numeric(tot_emp)) %>%
  rename(State = area_title) %>%
  select(State, RegNurse)

###

# Read lockdown data into R
# Number of days each state has been in lockdown
lkd.int = read.csv("countryLockdowndates.csv")

lkd.st = lkd.int %>%
  filter(Country.Region == "US") %>%
  select(Province, Date) %>%
  rename(State = Province) %>%
  mutate(Date = as.Date(Date, "%d/%m/%Y")) %>%
  mutate(LkdDuration =
           as.numeric(as.Date("2020-04-25")) - as.numeric(Date)) %>%
  mutate(LkdDuration = replace_na(LkdDuration, 0)) %>%
  select(State, LkdDuration)

###

# Read transportation data into R
# Number of unlinked passenger trips in thousands (2013)
transpo = read_xlsx("table_04-04_1.xlsx", skip = 3, n_max = 51, col_names = c("State", "drop", "Trips",

transpo.st = transpo %>%
  select(State, Trips) %>%
  rename(PubTrans = Trips)

###
```

```r
# Read race data into R
# Percentage of each race out of total population (2017)
demo.cty = read.csv("acs2017_county_data.csv")

demo.st = demo.cty %>%
  mutate(Hisp_Ct = ceiling(TotalPop*(Hispanic/100)),
         White_Ct = ceiling(TotalPop*(White/100)),
         Black_Ct = ceiling(TotalPop*(Black/100)),
         Native_Ct = ceiling(TotalPop*(Native/100)),
         Asian_Ct = ceiling(TotalPop*(Asian/100)),
         Pac_Ct = ceiling(TotalPop*(Pacific/100))) %>%
  select(State, TotalPop, Hisp_Ct, White_Ct,
         Black_Ct, Native_Ct, Asian_Ct, Pac_Ct) %>%
  group_by(State) %>%
  summarise_all(sum) %>%
  mutate(Hispanic = round((Hisp_Ct/TotalPop)*100, digits = 2),
         White = round((White_Ct/TotalPop)*100, digits = 2),
         Black = round((Black_Ct/TotalPop)*100, digits = 2),
         Native = round((Native_Ct/TotalPop)*100, digits = 2),
         Asian = round((Asian_Ct/TotalPop)*100, digits = 2),
         Pacific = round((Pac_Ct/TotalPop)*100, digits = 2)) %>%
  select(State, Hispanic, White,
         Black, Native, Asian, Pacific) %>%
  filter(State != "Puerto Rico")

###

# Read age data into R
# Percentage of population 65 and older (2018)
age = read.csv("PEP_2018_PEPAGESEX_with_ann.csv")

# Search for most recent census data
# age[,grep("2018sex0", names(age))]

age.st = age %>%
  select(GEO.display.label, est72018sex0_age999, est72018sex0_age65plus) %>%
  slice(3:53) %>%
  mutate(TotalPop = as.numeric(as.character(est72018sex0_age999)),
         OlderPop = as.numeric(as.character(est72018sex0_age65plus))) %>%
  mutate(Pct65Plus = round((OlderPop/TotalPop)*100, digits = 2)) %>%
  rename(State = GEO.display.label) %>%
  select(State, TotalPop, Pct65Plus)

###

# Read health insurance data into R
# Percentage of population uninsured (2018)
insurance = read_xlsx("Uninsured.xlsx", skip = 7, col_names = c("State", "Number", "Err1", "Percent", "E

insurance.st = insurance %>%
  select(State, Percent) %>%
  rename(Uninsured = Percent)
```

```r
###

# Read poverty data into R
# Percentage of population in poverty (Average 2016-2018)
poverty = read_xlsx("poverty.xlsx", skip = 8, col_names = c("State", "Percent", "Err"))

poverty.st = poverty %>%
  select(State, Percent) %>%
  rename(Poverty = Percent)

###

# Read unemployment data into R
# Percentage of population unemployed (March 2020)
unemployed = read_xlsx("unemployed.xlsx", skip = 1, col_names = c("State", "Percent", "Rank"))

unemployed.st = unemployed %>%
  arrange(State, desc(State)) %>%
  select(State, Percent) %>%
  rename(Unemp = Percent)

data = cv19.st %>%
  left_join(age.st, by = "State") %>%
  left_join(demo.st, by = "State") %>%
  left_join(lkd.st, by = "State") %>%
  left_join(nurse.st, by = "State") %>%
  left_join(hospital.st, by = "State") %>%
  left_join(transpo.st, by = "State") %>%
  left_join(insurance.st, by = "State") %>%
  left_join(poverty.st, by = "State") %>%
  left_join(unemployed.st, by = "State")
```

```
## Warning: Column `State` joining factors with different levels, coercing to
## character vector

## Warning: Column `State` joining character vector and factor, coercing into
## character vector

## Warning: Column `State` joining character vector and factor, coercing into
## character vector
```

```r
library(randomForest)
```

```
## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':
##
##     combine
```

```r
set.seed(123)
index <- sample(1:nrow(data), nrow(data)/(10/7))
```

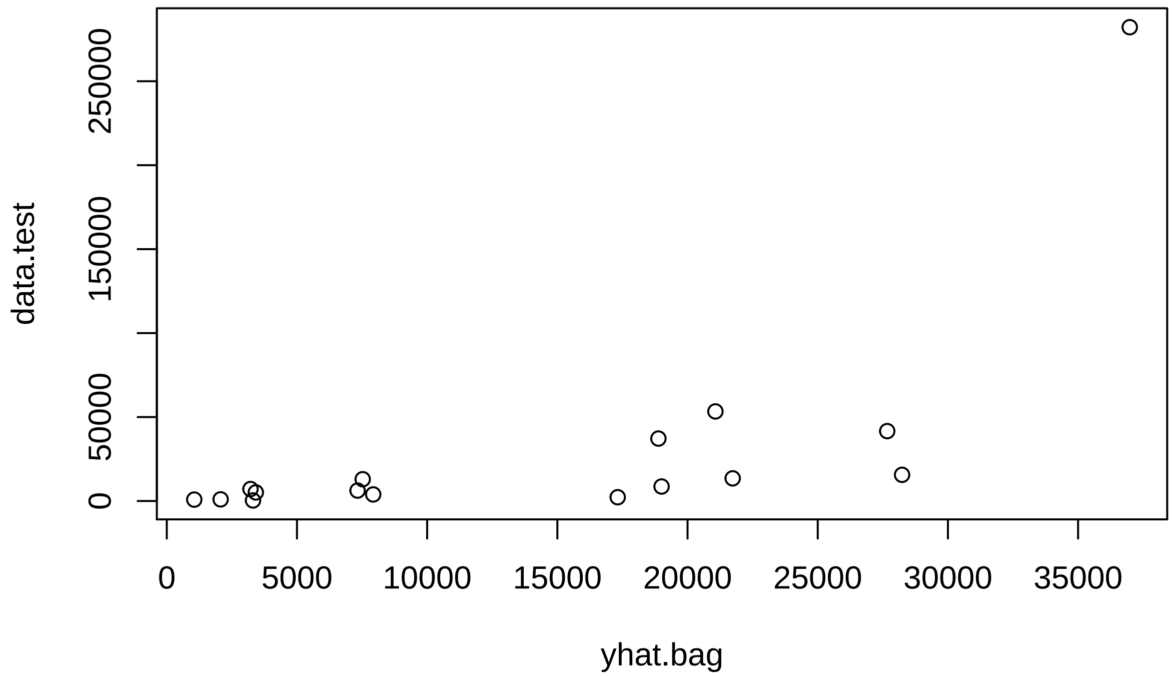
```r
covid.train <- data[index, ]
covid.test <- data[-index, ]

bag.data <- randomForest(Cases ~.-State-Deaths-PercentDeath, data=covid.train, mtry=15, importance=TRUE)
bag.data
```

```
##
## Call:
##  randomForest(formula = Cases ~ . - State - Deaths - PercentDeath,      data = covid.train, mtry = 15
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 15
##
##           Mean of squared residuals: 217089818
##                     % Var explained: 43.5
```

```r
importance(bag.data)
```

```
##               %IncMSE IncNodePurity
## TotalPop     8.732686     835301015
## Pct65Plus   -1.930895     102491633
## Hispanic     2.549272     104629462
## White        2.925415      89572899
## Black        3.143964     110859608
## Native       3.735335    6437081066
## Asian        3.006529     653556703
## Pacific      1.000422      17454143
## LkdDuration 11.147256    1584729276
## RegNurse     6.592137     812338271
## NumBeds      5.997396     750666252
## PubTrans     4.675696     691155268
## Uninsured    0.724092      54542550
## Poverty     -2.797856      60017020
## Unemp        1.934919      86175279
```

```r
##calculate the MSE for bagging model
yhat.bag <- predict(bag.data, newdata=covid.test)
data.test <- covid.test$Cases
plot(yhat.bag, data.test)
```




```r
MSE.bag <- mean((yhat.bag-data.test)^2)
print(MSE.bag)
```

```
## [1] 3895477905
```

```r
print(sqrt(MSE.bag))
```

```
## [1] 62413.76
```

Using the bagging method to build the model yields a Mean of squared residuals of 217089818 with the formula Cases ~ . - State - Deaths - PercentDeath. The percent of variance explained by the model is only 43.5%. The two most important variables in the model are LkdDuration and TotalPop. Using the test data we calculate a Test MSE = 3.8954779×10^9 with a square root of 6.2413764×10^4.

```r
#Build the random forest model using mtry = p/3 = 5
rf.data <- randomForest(Cases ~. -State-Deaths-PercentDeath, data=covid.train,
                        mtry=5, importance=TRUE)
rf.data
```

```
##
## Call:
##  randomForest(formula = Cases ~ . - State - Deaths - PercentDeath,      data = covid.train, mtry = 5
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 5
##
##           Mean of squared residuals: 236298994
##                     % Var explained: 38.5
```

```r
importance(rf.data)
```

```
##                %IncMSE IncNodePurity
## TotalPop     6.0580939    1237907107
## Pct65Plus    0.5112498     117098122
## Hispanic     1.7178776     147630580
```

```
## White        1.9858953     105856693
## Black        2.7771409     142114433
## Native       3.9085118    3013614482
## Asian        2.4351660     970728712
## Pacific     -0.6852457      46354874
## LkdDuration  9.1791248    2092901529
## RegNurse     7.6137048    1403111429
## NumBeds      7.8292993    1487748603
## PubTrans     5.3435991    1044470752
## Uninsured    0.9976111      88425435
## Poverty     -0.2446887     272674262
## Unemp        2.1693078     222205770
```

```
#calculate the MSE for the random Forest model.
yhat.rf <- predict(rf.data, newdata=covid.test)
MSE.rf <- mean((yhat.rf - data.test)^2)
print(MSE.rf)
```

```
## [1] 3752256854
```

```
print(sqrt(MSE.rf))
```

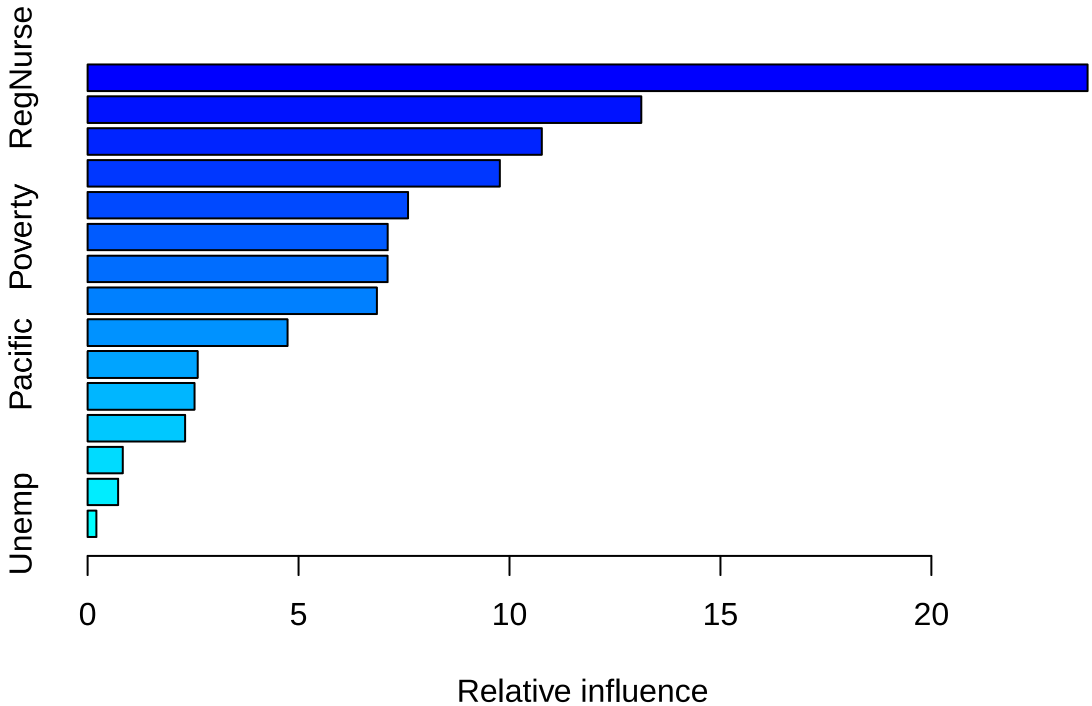
```
## [1] 61255.67
```

Using the Random Forest method to build the model yields a Mean of squared residuals of 236298994 with the formula Cases ~ . - State - Deaths - PercentDeath. The percent of variance explained by the model is only 38.5% which is lower than the bagging model (43.5). The two most important variables in the model are LkdDuration and NumBeds, with RegNurse coming in as a close third. Interestingly, LkdDuration was also the most important variable in the bagging model. Using the test data we calculate a Test MSE = 3.7522569×10^9 with a square root of 6.1255668×10^4 both slighlty lower than the MSE and root MSE for the bagging model.

```
#Build the boosting model for Cases
library(gbm)
```

```
## Loaded gbm 2.1.5
```

```
set.seed(123)
#n.minobsinnode is the min number of obs in trees' terminal nodes.
#bag.fraction is the fraction of the training set obs randomly selected to propose
#the next tree in the expansion. default is 0.5, increase if sample is small.
#since the data set is small, lower n.minobsinnode and increase bag.fraction
boost.data <- gbm(Cases ~. -State-Deaths-PercentDeath, data=covid.train,
                  distribution="gaussian", n.trees=5000, interaction.depth=4,
                  n.minobsinnode = 5, bag.fraction = 0.75)
summary(boost.data)
```




```
##                     var    rel.inf
## RegNurse       RegNurse 23.7026595
## TotalPop       TotalPop 13.1237952
## Asian             Asian 10.7664632
## LkdDuration LkdDuration  9.7714266
## Hispanic       Hispanic  7.5950651
## Poverty         Poverty  7.1123474
## Native           Native  7.1102398
## PubTrans       PubTrans  6.8572675
## NumBeds         NumBeds  4.7398792
## Pacific         Pacific  2.6100231
## Uninsured     Uninsured  2.5347888
## Black             Black  2.3111158
## Pct65Plus     Pct65Plus  0.8335478
## White             White  0.7231133
## Unemp             Unemp  0.2082677
```

```r
#Calculate the MSE for the boosting model.
yhat.boost <- predict(boost.data, newdata=covid.test, n.trees=5000)
MSE.boost <- mean((yhat.boost-data.test)^2)

print(MSE.boost)
```

```
## [1] 3253340773
```

```r
print(sqrt(MSE.boost))
```

```
## [1] 57038.06
```

A gradient boosted model with gaussian loss function was build using the formula Cases ~. -State-Deaths-PercentDeath. Five thousand iterations were performed with 15 predictors, all predictors had non-zero influence. The two most important variables in this model are RegNurse and TotalPop. LkdDuration was ranked 4th by the model. Using the test data we calculate a Test MSE $= 3.2533408 \times 10^9$ with a square root of 5.7038064×10^4. Both are lower than the MSE and root MSE for the random forest model, so of the three models Boosting performed the best.

Of the three models regressing number of cases of Covid-19 the Boosting model performed the best. However, the boosting model indicated the number of Registered Nurses as the most important predictor variable. This is difficult to explain as it is possible states with more registered nurses might provide more tests for Covid-19 and thus detect more cases (confounding?). All three models included lockdown duration as an important predictor variable. It was first for the bagging and random forest models, but only fourth for the Boosting model.