**Code Script:**

#Reading data here & preprocessing a little bit  
cases <- read.csv("cases.csv")  
demographic <- read.csv("demographic.csv")  
#get cases for one day for all countries and combine with factors  
onedate <- cases[which(cases$date == "2020-03-23"), ]  
covid <- data.frame(country = onedate$location, new\_cases = onedate$new\_cases)  
factors <- c(unique(demographic$Indicator)[2], unique(demographic$Indicator)[3], unique(demographic$Indicator)[5], unique(demographic$Indicator)[6], unique(demographic$Indicator)[11], unique(demographic$Indicator)[21], unique(demographic$Indicator)[22], unique(demographic$Indicator)[23], unique(demographic$Indicator)[35])  
  
dfList <- list()  
for(i in 1:length(factors)){  
  dfList[[i]] <- demographic[demographic$Indicator == factors[i], ]  
  dfList[[i]] <- dfList[[i]][which(dfList[[i]]$TIME == 2018),]  
  covid[[factors[i]]] <- dfList[[i]]$Value[match(covid$country, dfList[[i]]$Country)]  
}  
colnames(covid) <- c("country", "new\_cases", "lifeExp", "mortRate", "hiv", "ruralPop", "gdp", "oldPop", "youngPop", "midPop", "poverty")  
head(covid)  
  
  
######## splitting data into training and testing  
covid2 <- covid  
covid2 <- covid2[-which([is.na](http://is.na/)(covid2$lifeExp)),]  
covid2 <- covid2[-which([is.na](http://is.na/)(covid2$mortRate)),]  
covid2 <- covid2[-which([is.na](http://is.na/)(covid2$hiv)),]  
covid2 <- covid2[-which([is.na](http://is.na/)(covid2$new\_cases)),]  
set.seed(100)  # setting seed to reproduce results of random sampling  
trainingRowIndex <- sample(1:nrow(covid2), 0.8\*nrow(covid2))  # row indices for training data  
trainingData <- covid2[trainingRowIndex, ]  # model training data  
testData  <- covid2[-trainingRowIndex, ]   # test data  
  
##########################################  
############# LM  
lmMod <- lm(new\_cases ~ lifeExp + mortRate + hiv, data=trainingData)  # build the model  
distPred <- predict(lmMod, testData)  # predict distance  
summary (lmMod)  
  
plot(testData$new\_cases, distPred,  
     main="LM vs actual",  
     xlab="Actual")  
  
  
###############################################  
################## DT  
library(rpart)  
tree <- rpart::rpart(new\_cases ~ lifeExp + mortRate + hiv, data=trainingData)  
rpart.plot::rpart.plot(tree)  
rpart::printcp(tree)  
plotcp(tree)  
p <- predict(tree, trainingData)  
  
plot(trainingData$new\_cases, p,  
     main="LM vs actual",  
     xlab="Actual")  
  
###############################################  
###############################################  
################## NN  
require(nnet)  
range01 <- function(x){(x-min(x))/(max(x)-min(x))}  
covid3 <- covid2  
covid3$new\_cases <- range01(covid3$new\_cases)  
  
  
nnet.fit <- nnet(new\_cases ~ lifeExp + mortRate + hiv, data=covid3, size = 100)  
nnet.predict <- predict(nnet.fit)  
  
plot(covid3$new\_cases, nnet.predict,  
     main="Neural network predictions vs actual",  
     xlab="Actual")  
  
  
##############################################  
  
# Evaluating regression models  
# <https://medium.com/@amanbamrah/how-to-evaluate-the-accuracy-of-regression-results-b38e5512afd3>  
# 1. Mean Absolute Error (MAE) = 0 to Inf = Lower is better  
library(Metrics)  
actual <- testData$new\_cases  
predicted <- distPred  
mae(actual, predicted)  
# 2. Root Mean Squared Error (RMSE) = 0 to Inf = Lower is better = Always > MAE  
library(Metrics)  
actual <- testData$new\_cases  
predicted <- distPred  
rmse(actual, predicted)  
# 3. Mean Absolute Percentage Error (MAPE) = 0 to Inf = compare different models built on the same data = not suitable MAPE if data has small observations  
library(MLmetrics)  
actual <- testData$new\_cases  
predicted <- distPred  
MAPE(predicted, actual)  
# 4. R-Squared Score  
# 5. P-Value for covariates  
# 6. Plot actual vs predicted