SHORT_final

Dalit Hendel

12/10/2021

```
rm(list = ls(all.names = TRUE)) # will clear all objects, including hidden objects
gc() # free up memory and report memory usage
```

```
##          used (Mb) gc trigger (Mb) limit (Mb) max used (Mb)
## Ncells 459837 24.6     980361 52.4         NA   666917 35.7
## Vcells 856254  6.6    8388608 64.0      16384  1824099 14.0
```

```
# List of all packages
load.lib<-c("tidyverse", "rgdal", "raster","caret","sp",
"nnet","randomForest","kernlab","e1071", "dplyr", "doParallel")

# tidyverse - a collection of packages
# rgdal - the R GeoData Abstraction Layer (GDAL) -

# Loop through the packages, check if not installed, if true, install with dependencies.

install.lib<-load.lib[!load.lib %in% installed.packages()]
for(lib in install.lib) install.packages(lib,dependencies=TRUE)
sapply(load.lib,require,character=TRUE)
```

```
## Loading required package: tidyverse
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.1 ──
```

```
## ✓ ggplot2 3.3.5     ✓ purrr   0.3.4
## ✓ tibble  3.1.4     ✓ dplyr   1.0.7
## ✓ tidyr   1.1.4     ✓ stringr 1.4.0
## ✓ readr   2.0.2     ✓ forcats 0.5.1
```

```
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
## Loading required package: rgdal
```

```
## Loading required package: sp
```

```
## Please note that rgdal will be retired by the end of 2023,
## plan transition to sf/stars/terra functions using GDAL and PROJ
## at your earliest convenience.
##
## rgdal: version: 1.5-27, (SVN revision 1148)
## Geospatial Data Abstraction Library extensions to R successfully loaded
## Loaded GDAL runtime: GDAL 3.2.1, released 2020/12/29
## Path to GDAL shared files: /Library/Frameworks/R.framework/Versions/4.1/Resources/library/rgd
al/gdal
## GDAL binary built with GEOS: TRUE
## Loaded PROJ runtime: Rel. 7.2.1, January 1st, 2021, [PJ_VERSION: 721]
## Path to PROJ shared files: /Library/Frameworks/R.framework/Versions/4.1/Resources/library/rgd
al/proj
## PROJ CDN enabled: FALSE
## Linking to sp version:1.4-5
## To mute warnings of possible GDAL/OSR exportToProj4() degradation,
## use options("rgdal_show_exportToProj4_warnings"="none") before loading sp or rgdal.
## Overwritten PROJ_LIB was /Library/Frameworks/R.framework/Versions/4.1/Resources/library/rgda
l/proj
```

```
## Loading required package: raster
```

```
##
## Attaching package: 'raster'
```

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

```
## Loading required package: caret
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
## Loading required package: nnet
```

```
## Loading required package: 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
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
## Loading required package: kernlab
```

```
##
## Attaching package: 'kernlab'
```

```
## The following objects are masked from 'package:raster':
##
##     buffer, rotated
```

```
## The following object is masked from 'package:purrr':
##
##     cross
```

```
## The following object is masked from 'package:ggplot2':
##
##     alpha
```

```
## Loading required package: e1071
```

```
##
## Attaching package: 'e1071'
```

```
## The following object is masked from 'package:raster':
##
##     interpolate
```

```
## Loading required package: doParallel
```

```
## Loading required package: foreach
```

```
##
## Attaching package: 'foreach'
```

```
## The following objects are masked from 'package:purrr':
##
##     accumulate, when
```

```
## Loading required package: iterators
```

```
## Loading required package: parallel
```

```
##    tidyverse        rgdal       raster        caret           sp         nnet
##         TRUE         TRUE         TRUE         TRUE         TRUE         TRUE
## randomForest      kernlab       e1071        dplyr   doParallel
##         TRUE         TRUE         TRUE         TRUE         TRUE
```

```
# Load in the raster data of avgtemp
mtemp <- raster('wc2.1_10m_bio/wc2.1_10m_bio_1.tif')
class(mtemp) #raster
```

```
## [1] "RasterLayer"
## attr(,"package")
## [1] "raster"
```

```
crs(mtemp) #RS arguments: +proj=longlat +datum=WGS84 +no_defs
```

```
## Coordinate Reference System:
## Deprecated Proj.4 representation: +proj=longlat +datum=WGS84 +no_defs
## WKT2 2019 representation:
## GEOGCRS["WGS 84 (with axis order normalized for visualization)",
##     DATUM["World Geodetic System 1984",
##         ELLIPSOID["WGS 84",6378137,298.257223563,
##             LENGTHUNIT["metre",1]]],
##     PRIMEM["Greenwich",0,
##         ANGLEUNIT["degree",0.0174532925199433]],
##     CS[ellipsoidal,2],
##         AXIS["geodetic longitude (Lon)",east,
##             ORDER[1],
##             ANGLEUNIT["degree",0.0174532925199433,
##                 ID["EPSG",9122]]],
##         AXIS["geodetic latitude (Lat)",north,
##             ORDER[2],
##             ANGLEUNIT["degree",0.0174532925199433,
##                 ID["EPSG",9122]]]]
```

```
# loading in rain and elevation data
rain <- raster('wc2.1_10m_bio/wc2.1_10m_bio_12.tif')
elevation <- raster('wc2.1_10m_elev.tif')
```

```
# stacking raster data
rasStack = stack(elevation, mtemp, rain)
#creating a df of the raster info
df <- as.data.frame(rasStack, xy=TRUE)
df <- df[complete.cases(df), ] ##!!
#adding col names for variables that were layered
names(df) <- c('LONGITUDE', 'LATITUDE', 'ALTITUDE', 'ANNUAL_MEAN_TEMP', 'ANNUAL_PRECIPITATION')
#inspecting data
tail(df) #lots of NA data which is the water since this is only land temperatures
```

```
##         LONGITUDE  LATITUDE ALTITUDE ANNUAL_MEAN_TEMP ANNUAL_PRECIPITATION
## 2332795  179.0833 -89.91667     2634        -32.80713                    6
## 2332796  179.2500 -89.91667     2634        -32.81009                    6
## 2332797  179.4167 -89.91667     2634        -32.81202                    6
## 2332798  179.5833 -89.91667     2634        -32.81291                    6
## 2332799  179.7500 -89.91667     2634        -32.81253                    6
## 2332800  179.9167 -89.91667     2371        -31.62033                    7
```

```
dim(df)
```

```
## [1] 808053      5
```

```
# max min values of avg temp
maxValue(mtemp) # 30.98764
```

```
## [1] 30.98764
```

```
minValue(mtemp) # -54.72435
```

```
## [1] -54.72435
```

```
# max min values of rain
maxValue(rain) # 11191
```

```
## [1] 11191
```

```
minValue(rain) #0
```

```
## [1] 0
```

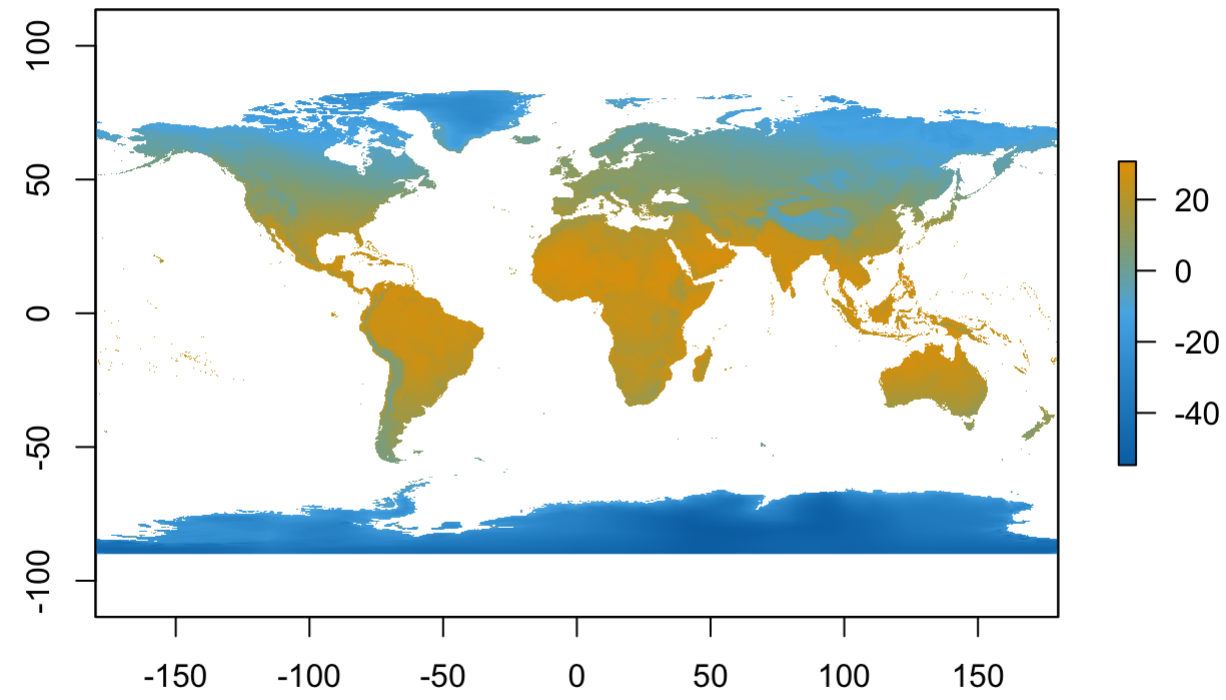
```
# max min values of elevation
maxValue(elevation) #6251
```

```
## [1] 6251
```

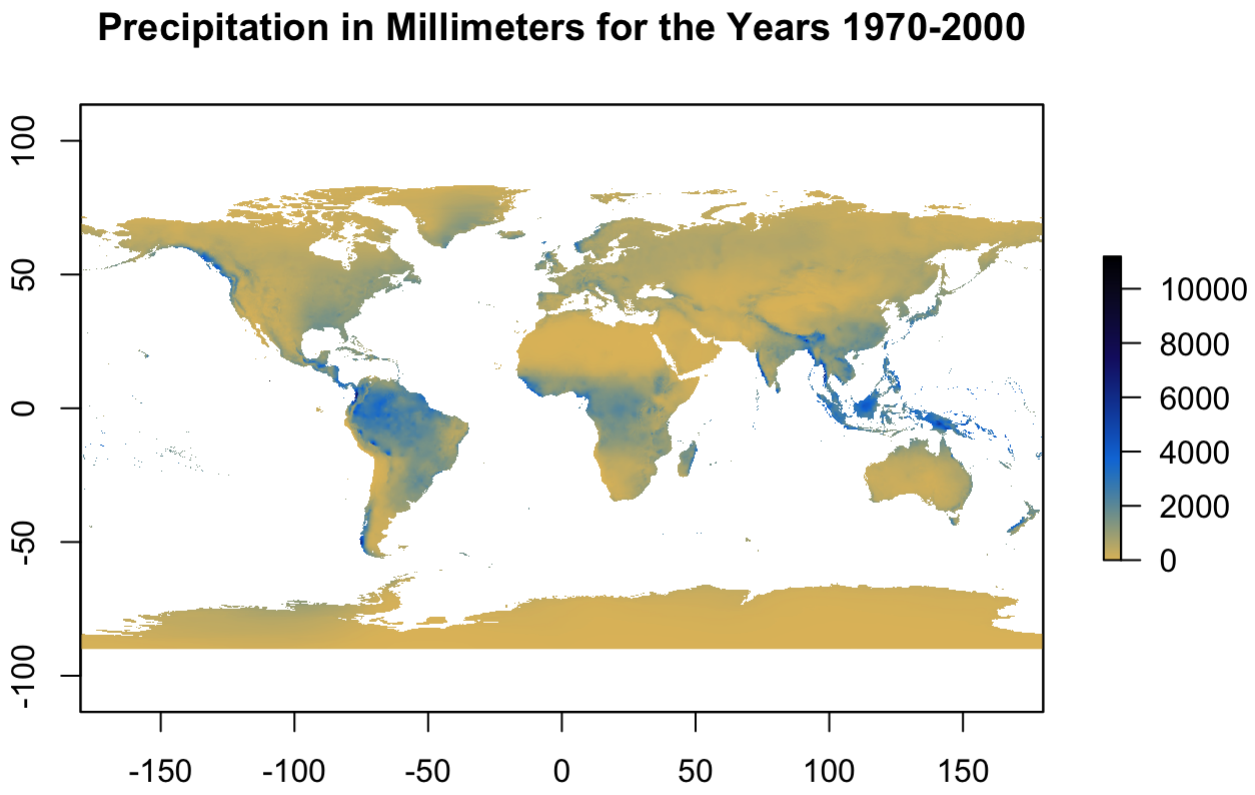
```
minValue(elevation) #-352
```

```
## [1] -352
```

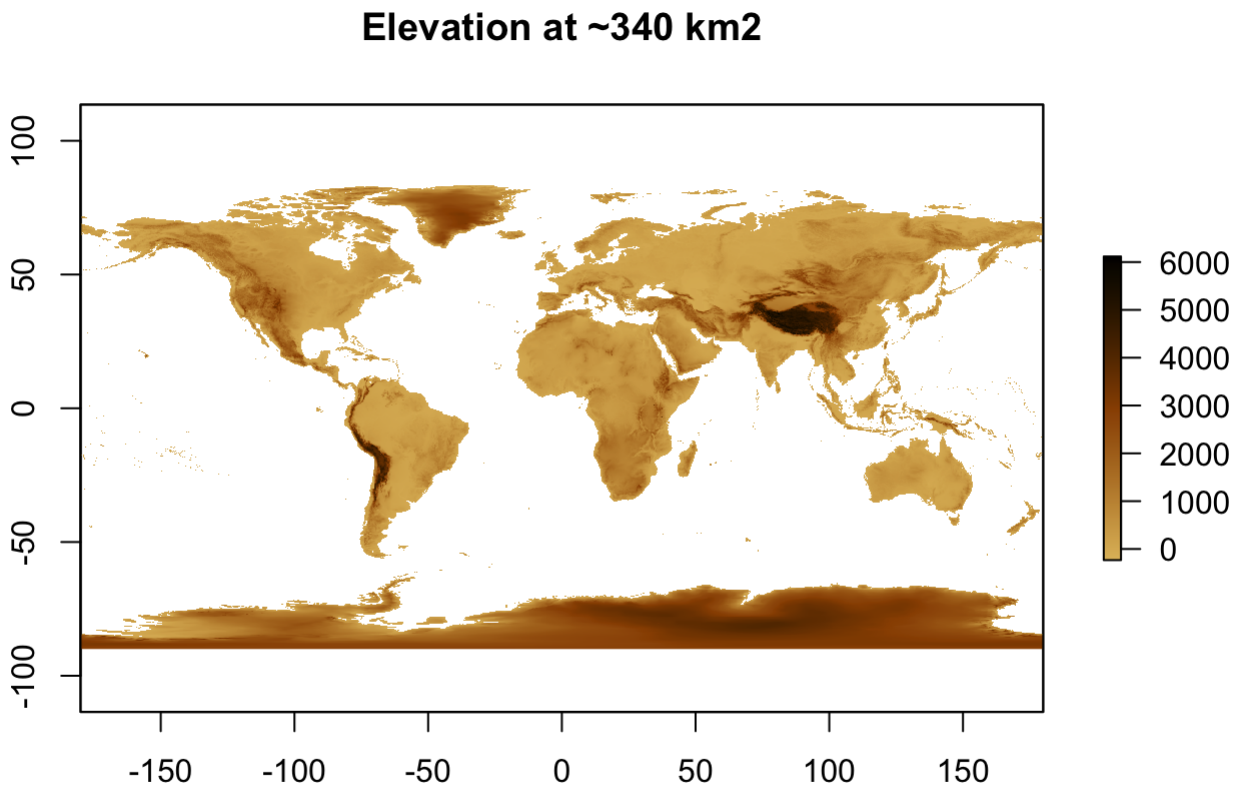
```
pal <- colorRampPalette(c('#0072B2', "#56B4E9", "#E69F00"))
plot(mtemp, col = pal(10000), main = 'Mean Temperature in Degrees Celsius for the Years 1970-200
0')
```




```
pa <- colorRampPalette(c("#E1BE6A", '#0C7BDC', 'midnightblue', 'black'))
plot(rain, col = pa(10000), main = 'Precipitation in Millimeters for the Years 1970-2000')
```




```
pa <- colorRampPalette(c("#E1BE6A", '#994F00', 'black'))
plot(elevation, col = pa(10000), main = 'Elevation at ~340 km2')
```




```
# this is our "samples" file with NA taken out AKA all the ocean locations removed
#df_historic <- df[complete.cases(df), ]
#head(df_historic)

#library(grid)
#us_states <- map_data('state')
#us_states <- us_states[,c(1,2)]
#colnames(us_states) <- c('LONGITUDE', 'LATITUDE')

#df <- left_join(us_states, df, by = c('LONGITUDE', 'LATITUDE'))
#dim(df)
```

```
# stacking raster data
rasStack = stack(elevation, mtemp, rain)
#creating a df of the raster info
df <- as.data.frame(rasStack, xy=TRUE)
#adding col names for variables that were layered
names(df) <- c('LONGITUDE', 'LATITUDE', 'ALTITUDE', 'ANNUAL_MEAN_TEMP', 'ANNUAL_PRECIPITATION')
#inspecting data
tail(df) #lots of NA data which is the water since this is only land temperatures
```

```
##         LONGITUDE  LATITUDE ALTITUDE ANNUAL_MEAN_TEMP ANNUAL_PRECIPITATION
## 2332795  179.0833 -89.91667     2634        -32.80713                    6
## 2332796  179.2500 -89.91667     2634        -32.81009                    6
## 2332797  179.4167 -89.91667     2634        -32.81202                    6
## 2332798  179.5833 -89.91667     2634        -32.81291                    6
## 2332799  179.7500 -89.91667     2634        -32.81253                    6
## 2332800  179.9167 -89.91667     2371        -31.62033                    7
```

```
dim(df)
```

```
## [1] 2332800       5
```

```
# do same with the future data
fmaxtemp126 <- raster('wc2.1_10m_tmax_BCC-CSM2-MR_ssp126_2021-2040/share/spatial03/worldclim/cmi
p6/7_fut/_/10m/BCC-CSM2-MR/ssp126/wc2.1_10m_tmax_BCC-CSM2-MR_ssp126_2021-2040.tif')
#fmaxtemp585 <- raster('wc2.1_10m_tmax_BCC-CSM2-MR_ssp585_2021-2040/share/spatial03/worldclim/cm
ip6/7_fut/_/10m/BCC-CSM2-MR/ssp585/wc2.1_10m_tmax_BCC-CSM2-MR_ssp585_2021-2040.tif')
# stacking raster data
rasStack = stack(fmaxtemp126)
#creating a df of the raster info
dff <- as.data.frame(rasStack, xy=TRUE)
#adding col names for variables that were layered
names(dff) <- c('LONGITUDE', 'LATITUDE', 'fmaxtemp126')
#inspecting data
head(dff) #lots of NA data which is the water since this is only land temperatures
```

```
##   LONGITUDE LATITUDE fmaxtemp126
## 1 -179.9167 89.91667          NA
## 2 -179.7500 89.91667          NA
## 3 -179.5833 89.91667          NA
## 4 -179.4167 89.91667          NA
## 5 -179.2500 89.91667          NA
## 6 -179.0833 89.91667          NA
```

```
dim(dff)
```

```
## [1] 2332800       3
```

```
dff <- dff[complete.cases(dff), ]
head(dff)
```

```
##       LONGITUDE LATITUDE fmaxtemp126
## 82927 -38.91667 83.58333       4.500
## 82928 -38.75000 83.58333       4.500
## 82929 -38.58333 83.58333      -0.625
## 82930 -38.41667 83.58333     -12.575
## 82931 -38.25000 83.58333     -24.700
## 82934 -37.75000 83.58333     -24.725
```

```
# max and min
maxValue(fmaxtemp126) # 42.98125
```

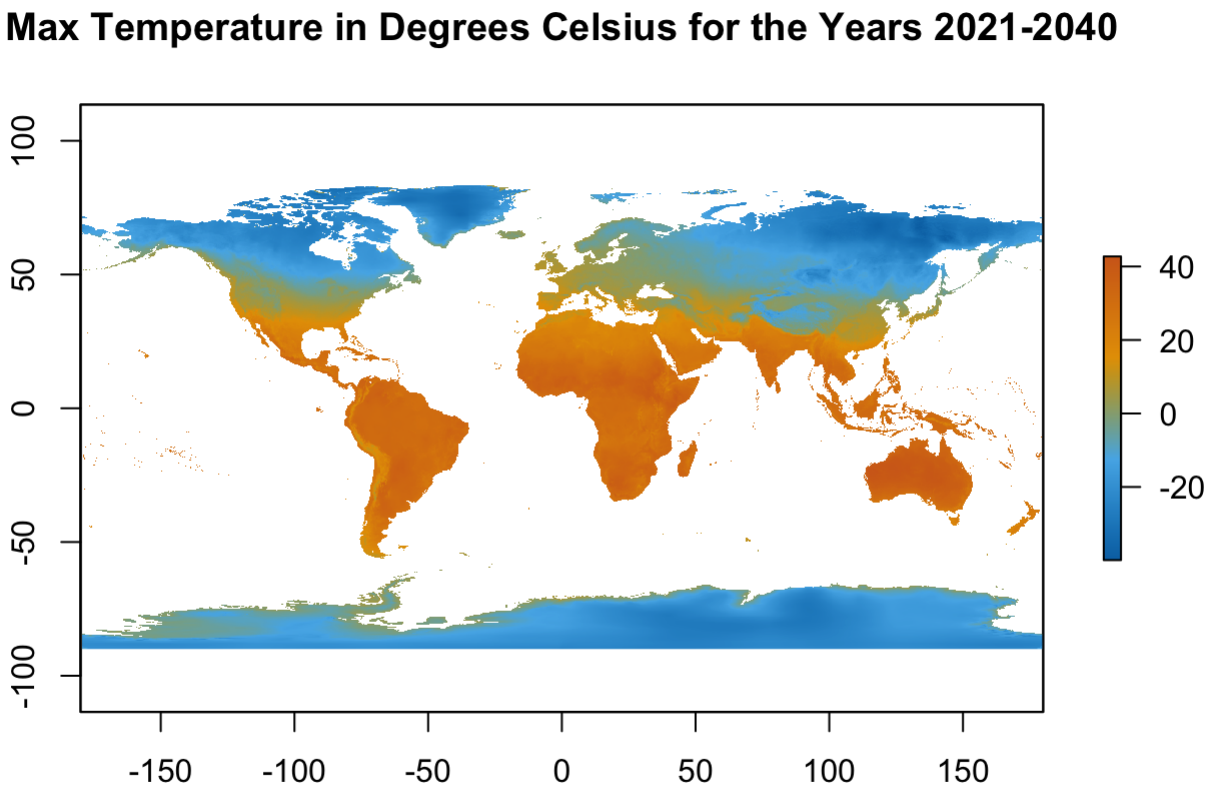
```
## [1] 42.98125
```

```
minValue(fmaxtemp126) # -40.1625
```

```
## [1] -40.1625
```

```
#maxValue(fmaxtemp585) # 43.88125
#minValue(fmaxtemp585) # -40.1625
```

```
pal <- colorRampPalette(c('#0072B2', "#56B4E9", "#E69F00", 'chocolate'))
plot(fmaxtemp126, col = pal(100000), main = 'Max Temperature in Degrees Celsius for the Years 20
21-2040')
```




```
# now find the deltas for past vs future to see the areas that have the greatest delta (past the limit set by pairs accord maybe?)
#merge dfs that have NAS and then subtract 2 cols
CLIMATEdf <- inner_join(df, dff, by = c('LONGITUDE','LATITUDE'))
tail(CLIMATEdf) #### WHY DO THEY HAVE DIFFERNT NA VALUES IN THE FILE
```

```
##        LONGITUDE LATITUDE ALTITUDE ANNUAL_MEAN_TEMP ANNUAL_PRECIPITATION
## 805688  179.0833   -89.75     2794        -32.80704                    7
## 805689  179.2500   -89.75     2794        -32.80835                    7
## 805690  179.4167   -89.75     2794        -32.80852                    6
## 805691  179.5833   -89.75     2794        -32.80915                    6
## 805692  179.7500   -89.75     2794        -32.81005                    7
## 805693  179.9167   -89.75     2514        -30.99528                    6
##        fmaxtemp126
## 805688     -12.800
## 805689     -12.725
## 805690     -12.700
## 805691     -12.700
## 805692     -12.700
## 805693     -11.825
```

```
Climate_dfs <- CLIMATEdf[complete.cases(CLIMATEdf), ]
head(Climate_dfs)
```

```
##   LONGITUDE LATITUDE ALTITUDE ANNUAL_MEAN_TEMP ANNUAL_PRECIPITATION fmaxtemp126
## 1 -38.91667 83.58333        0         0.000000                  115       4.500
## 2 -38.75000 83.58333        0         0.000000                  115       4.500
## 3 -38.58333 83.58333        0        -2.592391                  116      -0.625
## 4 -38.41667 83.58333        0        -8.346475                  115     -12.575
## 5 -38.25000 83.58333        0       -16.416666                  115     -24.700
## 6 -37.75000 83.58333        0       -17.895636                  120     -24.725
```

```
#now sub future - past on temp and make the col diff
Climate_dfs$diff <- Climate_dfs$fmaxtemp126 - Climate_dfs$ANNUAL_MEAN_TEMP
head(Climate_dfs)
```

```
##   LONGITUDE LATITUDE ALTITUDE ANNUAL_MEAN_TEMP ANNUAL_PRECIPITATION fmaxtemp126
## 1 -38.91667 83.58333        0         0.000000                  115       4.500
## 2 -38.75000 83.58333        0         0.000000                  115       4.500
## 3 -38.58333 83.58333        0        -2.592391                  116      -0.625
## 4 -38.41667 83.58333        0        -8.346475                  115     -12.575
## 5 -38.25000 83.58333        0       -16.416666                  115     -24.700
## 6 -37.75000 83.58333        0       -17.895636                  120     -24.725
##        diff
## 1  4.500000
## 2  4.500000
## 3  1.967391
## 4 -4.228526
## 5 -8.283335
## 6 -6.829365
```

```
max(Climate_dfs$diff) #28.5719
```

```
## [1] 28.5719
```

```
min(Climate_dfs$diff) #-25.80246
```

```
## [1] -25.80246
```

```
mean(Climate_dfs$diff) #3.414988
```

```
## [1] 3.414988
```

```
str(Climate_dfs)
```

```
## 'data.frame':    805693 obs. of  7 variables:
##  $ LONGITUDE           : num  -38.9 -38.8 -38.6 -38.4 -38.2 ...
##  $ LATITUDE            : num  83.6 83.6 83.6 83.6 83.6 ...
##  $ ALTITUDE            : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ ANNUAL_MEAN_TEMP    : num  0 0 -2.59 -8.35 -16.42 ...
##  $ ANNUAL_PRECIPITATION: num  115 115 116 115 115 120 120 121 121 123 ...
##  $ fmaxtemp126         : num  4.5 4.5 -0.625 -12.575 -24.7 ...
##  $ diff                : num  4.5 4.5 1.97 -4.23 -8.28 ...
```

```
dim(Climate_dfs)
```

```
## [1] 805693      7
```

```
Climate_df <- Climate_dfs[c(1:5000),]
```

```
# add a variable that says if the location is above or below the threshold
# is it okay to use UN level???
Climate_df <- Climate_df %>%
  mutate(class = case_when(
    (abs(Climate_df$diff) > 1.5) ~ 1,
    (abs(Climate_df$diff) <= 1.5) ~ 2)) # class == 1 means large climate change diff area #class
== 0 means not past threshold in delta
table(Climate_df$class)
```

```
##
##    1    2
## 4868  132
```

```
#ASK
```

```
#removing columns we dont want in the ML
keeps <- c("LONGITUDE", "LATITUDE", "ALTITUDE", "ANNUAL_MEAN_TEMP", "ANNUAL_PRECIPITATION", "cla
ss")
samples <- Climate_df[keeps]
is.data.frame(samples) #TRUE
```

```
## [1] TRUE
```

```
head(samples)
```

```
##   LONGITUDE LATITUDE ALTITUDE ANNUAL_MEAN_TEMP ANNUAL_PRECIPITATION class
## 1 -38.91667 83.58333        0         0.000000                  115     1
## 2 -38.75000 83.58333        0         0.000000                  115     1
## 3 -38.58333 83.58333        0        -2.592391                  116     1
## 4 -38.41667 83.58333        0        -8.346475                  115     1
## 5 -38.25000 83.58333        0       -16.416666                  115     1
## 6 -37.75000 83.58333        0       -17.895636                  120     1
```

```
# attempt at random forest
## 70% of the sample size 30% test size
smp <- floor(0.70 * nrow(samples))
#
set.seed(42)
train_ind <- sample(seq_len(nrow(samples)), size = smp)

trn <- samples[train_ind, ]
tst <- samples[-train_ind, ]
head(trn)
```

```
##      LONGITUDE LATITUDE ALTITUDE ANNUAL_MEAN_TEMP ANNUAL_PRECIPITATION class
## 2609 -53.25000 81.91667        0        -16.09488                  101     1
## 4069 -32.25000 81.41667      341        -18.60710                  219     1
## 2369 -24.91667 82.08333        0          0.00000                  164     1
## 1098 -68.08333 82.58333      419        -17.59305                  122     1
## 1252 -28.75000 82.58333      589        -20.24627                  248     1
## 634  -37.91667 82.91667       27        -17.48000                  125     1
```

```
# Set up a resampling method in the model training process from lab
#Let's create a cluster

doParallel::registerDoParallel((parallel::detectCores()-2))

tc <- caret::trainControl(method = "repeatedcv", # repeated cross-validation of the training dat
a
                   number = 10, # number of folds
                   repeats = 5, # number of repeats
                   allowParallel = TRUE, # allow use of multiple cores if specified in training
                   verboseIter = TRUE) # view the training iterations

##
nnet.grid = expand.grid(size = seq(from = 2, to = 10, by = 2), # number of neurons units in the
 hidden layer

                        decay = seq(from = 0.1, to = 0.5, by = 0.1)) # regularization parameter
 to avoid over-fitting


rf.grid <- expand.grid(mtry=1:20) # number of variables available for splitting at each tree nod
e

#svm.grid <- expand.grid(sigma=seq(from = 0.01, to = 0.10, by = 0.02), # controls for non-linear
ity in the hyperplane
#                         C=seq(from = 2, to = 10, by = 2)) # controls the influence of each supp
ort vector
```

```
# Train the neural network model
nnet_model <- caret::train(x = trn[,(4:ncol(trn)-1)], y = as.factor(as.integer(as.factor(trn$cla
ss))),
                    method = "nnet", metric="Accuracy", trainControl = tc, tuneGrid = nnet.grid)
```

```
## # weights:  41
## initial  value 1153.712065
## iter  10 value 484.555614
## iter  20 value 461.201288
## iter  30 value 422.688499
## iter  40 value 396.821672
## iter  50 value 365.749404
## iter  60 value 335.780138
## iter  70 value 333.817922
## iter  80 value 320.479188
## iter  90 value 313.017017
## iter 100 value 300.729884
## final  value 300.729884
## stopped after 100 iterations
```

```
# Train the random forest model
rf_model <- caret::train(x = trn[,(4:ncol(trn)-1)], y = as.factor(as.integer(as.factor(trn$clas
s))),
                     method = "rf", metric="Accuracy", trainControl = tc, tuneGrid = rf.grid)
```

```
#turn samples into a raster file
r <- rasterFromXYZ(samples)
```

```
# Apply the neural network model to the Sentinel-2 data.
nnet_prediction = raster::predict(r, model=nnet_model)

# Apply the random forest model to the Sentinel-2 data
rf_prediction = raster::predict(r, model=rf_model)

# Apply the support vector machines model to the Sentinel-2 data
#svm_prediction = raster::predict(r, model=svm_model)

# Convert the evaluation data into a spatial object using the X and Y coordinates and extract pr
edicted values
tst.sp = SpatialPointsDataFrame(coords = cbind(tst$LONGITUDE, tst$LATITUDE), data = tst,
                                proj4string = crs("+proj=utm +zone=33 +datum=WGS84 +units=m +no_
defs +ellps=WGS84 +towgs84=0,0,0"))
```

```
## Superimpose evaluation points on the predicted classification and extract the values
# neural network
nnet_Eval = raster::extract(nnet_prediction, tst.sp)
# random forest
rf_Eval = raster::extract(rf_prediction, tst.sp)
# support vector machines
#svm_Eval = raster::extract((svm_prediction), tst.sp)

# Create an error matrix for each of the classifiers
nnet_errorM = confusionMatrix(as.factor(nnet_Eval),as.factor(tst$class)) # nnet is a poor classi
fier, so it will not capture all the classes
rf_errorM = confusionMatrix(as.factor(rf_Eval),as.factor(tst$class))
##!! svm_errorM = confusionMatrix(as.factor(svm_Eval),as.factor(tst$class))

paste0("  Neural Net accuracy:  ", round(nnet_errorM$overall[1],2))
```

```
## [1] "  Neural Net accuracy:  0.97"
```

```
paste0("  Random Forest accuracy:  ", round(rf_errorM$overall[1],2))
```

```
## [1] "  Random Forest accuracy:  0.98"
```

```
## !! paste0("  SVM accuracy:  ", round(svm_errorM$overall[1],2))
```

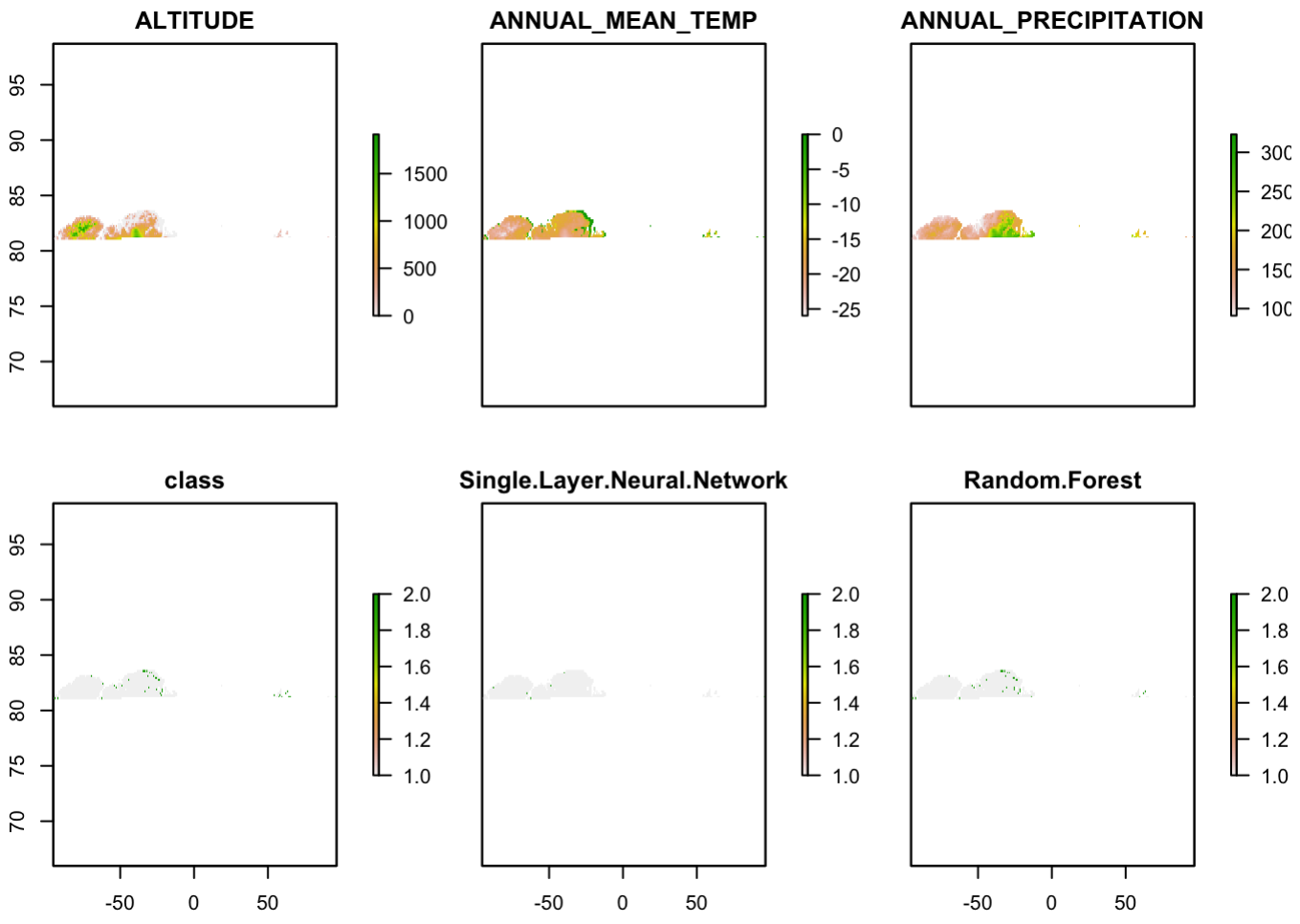
```
print(nnet_errorM$table)
```

```
##           Reference
## Prediction    1    2
##          1 1460   36
##          2    2    2
```

```
print(rf_errorM$table)
```

```
##           Reference
## Prediction    1    2
##          1 1453   27
##          2    9   11
```

```
# Plot the results next to one another along with the 2018 NMD dataset for comparison
crs(r) <- crs(nnet_prediction) # Correct the coordinate reference system so it matches with the
 rest
rstack = stack(r, nnet_prediction, rf_prediction) # combine the layers into one stack
names(rstack) = c("ALTITUDE", "ANNUAL_MEAN_TEMP", "ANNUAL_PRECIPITATION", "class", "Single Layer
Neural Network", "Random Forest") # name the stack
plot(rstack) # plot it!
```



```
doParallel::stopImplicitCluster()
#https://sites.tufts.edu/datalab/covid-19-data-lab-tools-preparing-to-work-remotely/
```