Classify a raster in R.

Learning Objectives

After completing this tutorial, you will be able to:

- Crop a raster dataset in R using a vector extent object derived from a shapefile.
- Open a shapefile in R.

What you need

You need `R` and `RStudio` to complete this tutorial. Also you should have an `earth-analytics` directory setup on your computer with a `/data` directory with it.

- How to Setup R / RStudio
- Setup your working directory
- Intro to the R & RStudio Interface

R Libraries to Install:

- **raster:** `install.packages("raster")`
- **rgdal:** `install.packages("rgdal")`

If you have not already downloaded the week 3 data, please do so now. Download Week 3 Data (~250 MB){:data-proofer-ignore=" .btn }

In this lesson, we will learn how to crop a raster dataset in `R`. Previously, we reclassified a raster in R, however the edges of our raster dataset were uneven. In this lesson, we will learn how to crop a raster - to create a new raster object / file that we can share with colleagues and / or open in other tools such as QGIS.

Load libraries

```
# load the raster and rgdal libraries
library(raster)
library(rgdal)
```

Open raster and vector layers

First, we will use the `raster()` function to open a raster layer. Let's open the canopy height model that we created in the previous lesson

```
# open raster layer
lidar_chm <- raster("data/week3/BLDR_LeeHill/outputs/lidar_chm.tif")

# plot CHM
plot(lidar_chm,
     col=rev(terrain.colors(50)))
```
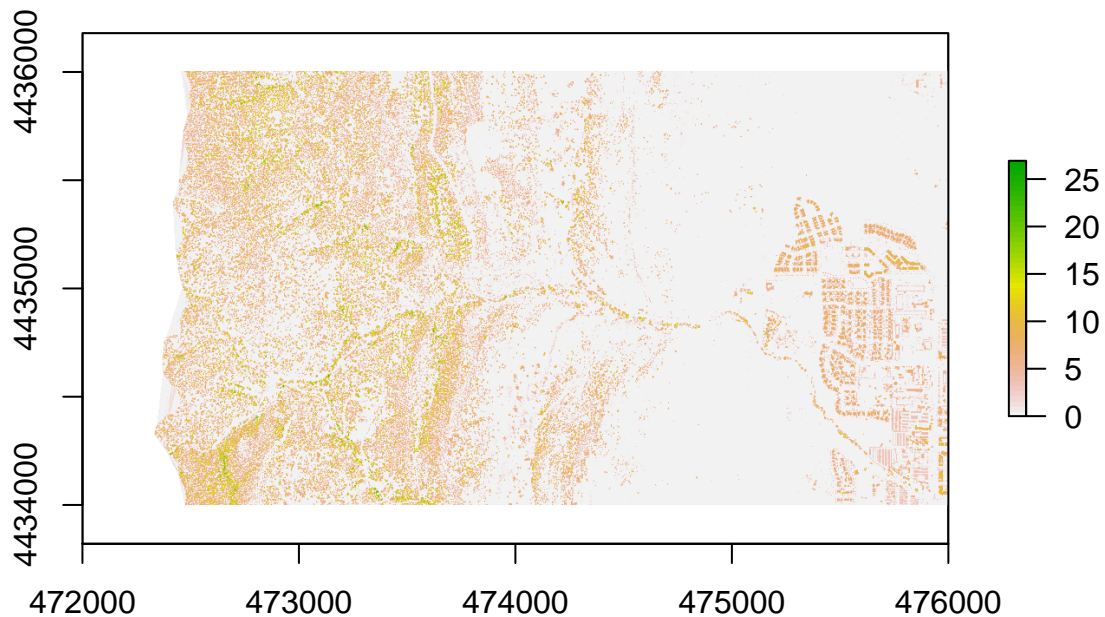
Figure 1: lidar chm plot

Open vector layer

Next, let's open up a vector layer that contains the crop extent that we want to use to crop our data. To open a shapefil we use the `readOGR()` function.

`readOGR()` requires two components:

1. The directory where our shapefile lives: data/week3/BLDR_LeeHill/
2. The name of the shapefile (without the extension): clip-extent

```
# import the vector boundary
crop_extent <- readOGR("data/week3/BLDR_LeeHill/",
                       "clip-extent")
## OGR data source with driver: ESRI Shapefile
## Source: "data/week3/BLDR_LeeHill/", layer: "clip-extent"
## with 1 features
## It has 1 fields
## Integer64 fields read as strings:  id


# plot imported shapefile
# notice that we use add=T to add a layer on top of an existing plot in R.
plot(crop_extent,
     main="Shapefile imported into R - crop extent",
     axes=T,
     border="blue")
```

Now that we have imported the shapefile. We can use the crop() function in R to crop the raster data using the vector shapefile.

```
# crop the lidar raster using the vector extent
lidar_chm_crop <- crop(lidar_chm, crop_extent)
plot(lidar_chm_crop, main="Cropped lidar chm")

# add shapefile on top of the existing raster
```
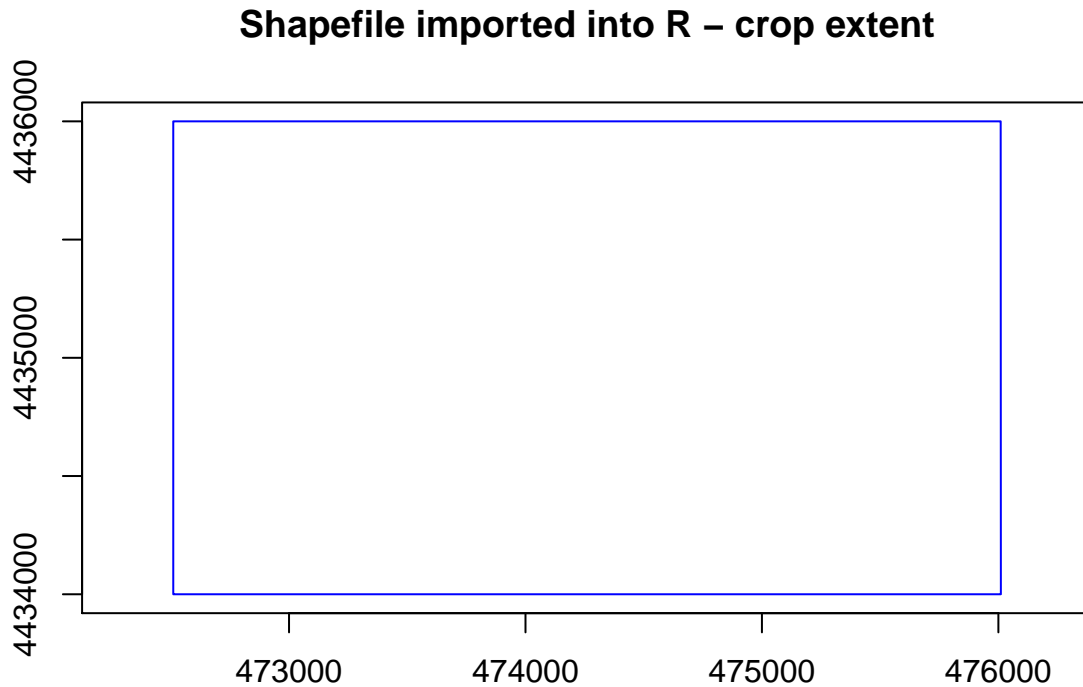
Shapefile imported into R – crop extent

Figure 2: shapefile crop extent plot

```r
plot(crop_extent, add=T)
```

Challenge - crop change over time layers

In the previous lesson, you created 2 plots: 1. A classified raster map that shows **positive and negative change** in the canopy height model before and after the flood. To do this you will need to calculate the difference between two canopy height models. 2. A classified raster map that shows **positive and negative change** in terrain extracted from the pre and post flood Digital Terrain Models before and after the flood.

Create the same two plots except this time CROP each of the rasters that you plotted using the shapefile: `data/week3/BLDR_LeeHill/crop_extent.shp`

For each plot, be sure to:

- Add a legend that clearly shows what each color in your classified raster represents
- Use better colors that I used in my example above!.
- Add a title to your plot.

You will include these plots in your final report due next week.

Check out this cheatsheet for more on colors in `R`.

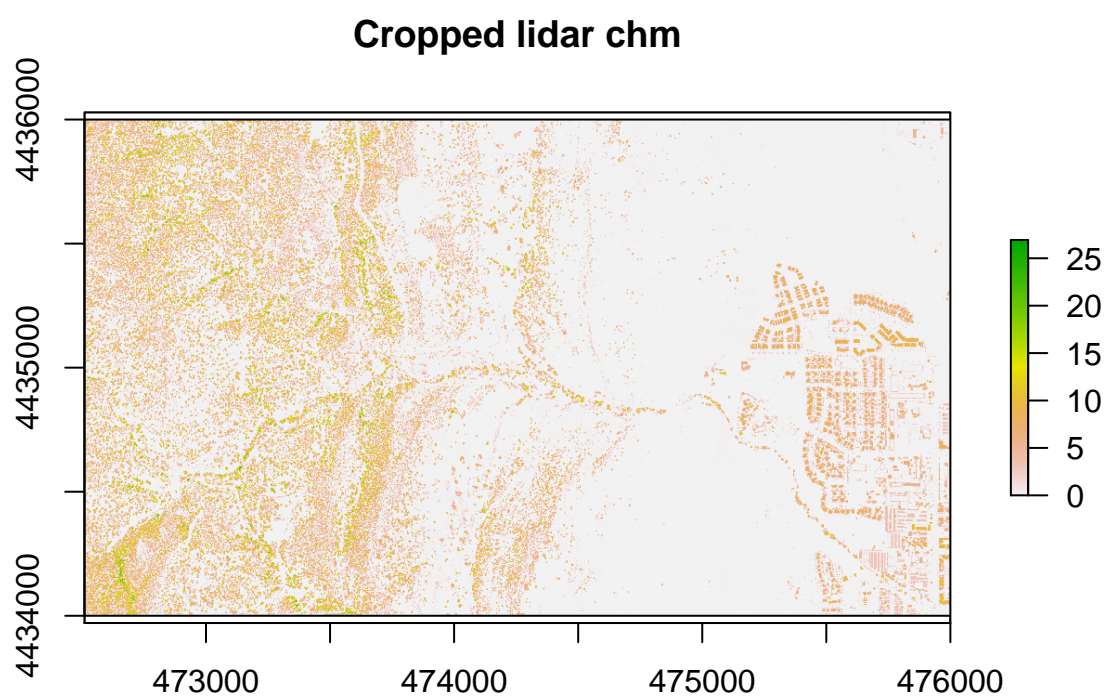Or type: `grDevices::colors()` into the r console for a nice list of colors!

Figure 3: lidar chm cropped with vector extent on top