Landcover Classification using Satellite Data

Owen Xu Li

April 2024

Introduction

Satellite data can be used to estimate the type of landcover at locations around the world. This approach can be an time and cost effective alternative to manually inspecting these locations in person. In this report, we will explore a data set containing satellite data and manually labeled landcover types for locations in Benin. We will use this data to build a model that predicts the landcover type at a location based on the satellite data.

Landcover types:

1. *Built-up:* Built-up areas, also known as urban areas, are regions where the landscape is dominated by human-made structures such as buildings, roads, and other infrastructure. These areas are characterized by high-density development and are typically associated with cities, towns, and other developed communities. Built-up areas include residential, commercial, industrial, and institutional land uses.
2. *Cropland:* Cropland refers to land that is used for the cultivation of crops. This landcover type includes fields used for growing a variety of crops such as grains, vegetables, fruits, and other agricultural products. Cropland can vary in size from small family farms to large-scale industrial agricultural operations.
3. *Natural Forest:* Natural forests are areas covered by trees and other vegetation that have developed through natural processes without significant human intervention. These forests play a crucial role in maintaining biodiversity, regulating climate, and providing habitat for wildlife. Natural forests can be found in a variety of climatic zones, from tropical rainforests to temperate and boreal forests.
4. *Orchard:* An orchard is a type of agricultural land where trees or shrubs are cultivated primarily for fruit production. Orchards are typically designed for intensive farming practices, focusing on high yields of specific fruit species such as apples, oranges, cherries, and nuts. These areas require careful management and maintenance to ensure healthy tree growth and abundant fruit production.

Data

The data `labeled_points.Rdata` contains data on blocks of land in Benin.

```r
load('data/labeled_points.Rdata')
```

The file contains two data frames.

1. labeled. The object `labeled` has 400 locations (with unique identifier `ID`). The `landcover` type at each location has been manually labeled by a human. Each `ID` has a unique latitude (`lat`) and longitude (`lon`) and can be thought of as a pixel in an image.

```r
head(labeled) %>%
  as.data.frame()
```

```
##      ID      lat      lon landcover
## 1 17394 11.17707 2.297213   builtup
```

```
## 2 15545 11.11946 2.853562   builtup
## 3 15722 11.12590 2.940327   builtup
## 4 15489 11.11835 2.854627   builtup
## 5 10946 10.98139 3.283267   builtup
## 6  5208 10.78519 2.806184   builtup
```

```r
unique(labeled$landcover)
```

```
## [1] "builtup"   "cropland"  "natforest" "orchard"
```

We see that the four labels are `builtup`, `cropland`, `natforest`, and `orchard`.

2. labeled_train. The object `labeled_train` has 3 years of satellite imagery for each `ID`. Images were collected every 16 days, and the `year`, `month`, `day`, and `date` for each location are given in the data. These were all taken by the Landsat 7 satellite. The other columns are

- **ID.** Unique identifier for the location, same as in `labeled`.
- **B1 to B8.** These are 8 bands from the image, including a red band, green band, blue band, infrared band, etc. These measure the strength of the red, green, and blue wavelengths in an image as well as the strength of other wavelengths on the electromagnetic spectrum that are not visible to the human eye. You can find more information about these on the Landsat 7 page.
- **NDVI.** Normalized Difference Vegetation Index. The NDVI is a common index used for summarizing satellite image data. According to its Wikipedia page, NDVI "is a simple graphical indicator that can be used to analyze remote sensing measurements, often from a space platform, assessing whether or not the target being observed contains live green vegetation." Note that the formula given on that page is

$$NDVI = \frac{NIR - Red}{NIR + Red}$$

  The data we use contains both the NIR (near-infrared) and Red bands (bands 4 and 3, respectively, according to https://www.usgs.gov/landsat-missions/landsat-7).
- **NDBI.** Normal Difference Built-up Index. Similar to NDVI, but for detecting built-up areas.
- **EVI.** Enhanced Vegetation Index. Like NDVI, but performs better under some conditions.

```r
labeled_train %>%
  as.data.frame() %>%
  head()
```

```
##   B1 B2 B3 B4  B5 B6_VCID_1 B6_VCID_2 B7 B8      lat      lon year month day
## 1 66 60 73 68  93       146       176 76 69 11.16359 3.413116 2017     1   3
## 2 64 59 69 72  89       147       178 72 69 11.16244 3.414022 2017     1   3
## 3 66 60 74 70  93       148       180 79 72 11.01509 3.416367 2017     1   3
## 4 69 63 85 75 113       151       185 90 75 10.93011 3.553716 2017     1   3
## 5 70 63 79 71 101       149       182 83 73 10.93055 3.553230 2017     1   3
## 6 62 55 66 68  83       148       181 69 67 10.92047 3.486705 2017     1   3
##         date    ID        NDVI       NDBI        EVI
## 1 2017-01-03 16993 -0.03546099 0.15527950 -1.0416667
## 2 2017-01-03 16894  0.02127660 0.10559006  1.0714286
## 3 2017-01-03 11728 -0.02777778 0.14110429 -0.5000000
## 4 2017-01-03  9560 -0.06250000 0.20212766 -0.3649635
## 5 2017-01-03  9574 -0.05333333 0.17441860 -0.9523810
## 6 2017-01-03  8938  0.01492537 0.09933775        Inf
```

These band values will be different depending on the landcover type.

The following are known relationships:

- NDVI is known to be a very good indicator of vegetation
- The band values show seasonal trends, since landcover can show seasonal changes (e.g. trees lose their leaves in the fall)

- The peaks and troughs can be shifted in time for different landcover types (different types of vegetation peak at different times).
- The difference between peaks and troughs can vary among landcover types.

Data Preparation

We first join our `labeled` and `labeled_train` data sets on `ID`.

```
labeled = labeled %>%
  select(ID, landcover)

d = labeled_train %>%
  left_join(labeled, by = 'ID')
```

Finally, let's add a column for vegetation called `veg` that is 1 if the `landcover` is `natforest`, `orchard`, or `cropland`, and 0 otherwise. We'll also add a column for built-up called `builtup` that is 1 if the `landcover` is `builtup`, and 0 otherwise. Thus, we are adding indicators for vegetation and built-up areas.

```
d = d %>%
  mutate(veg = ifelse(landcover %in% c('natforest', 'orchard', 'cropland'), 1, 0),
         builtup = ifelse(landcover == 'builtup', 1, 0),
         EVI = ifelse(is.infinite(EVI), NA, EVI))

glimpse(d)
```

```
## Rows: 18,308
## Columns: 22
## $ B1        <dbl> 66, 64, 66, 69, 70, 62, 64, 65, 67, 66, 63, 63, 69, 58, 60, ~
## $ B2        <dbl> 60, 59, 60, 63, 63, 55, 60, 59, 57, 58, 55, 57, 63, 50, 49, ~
## $ B3        <dbl> 73, 69, 74, 85, 79, 66, 79, 69, 68, 67, 64, 68, 81, 53, 58, ~
## $ B4        <dbl> 68, 72, 70, 75, 71, 68, 72, 65, 64, 64, 65, 63, 72, 82, 84, ~
## $ B5        <dbl> 93, 89, 93, 113, 101, 83, 100, 86, 88, 79, 78, 80, 97, 69, 7~
## $ B6_VCID_1 <dbl> 146, 147, 148, 151, 149, 148, 150, 149, 148, 149, 148, 150, ~
## $ B6_VCID_2 <dbl> 176, 178, 180, 185, 182, 181, 183, 182, 179, 181, 180, 182, ~
## $ B7        <dbl> 76, 72, 79, 90, 83, 69, 87, 72, 76, 68, 64, 69, 78, 43, 45, ~
## $ B8        <dbl> 69, 69, 72, 75, 73, 67, 74, 65, 66, 65, 63, 65, 77, 68, 72, ~
## $ lat       <dbl> 11.16359, 11.16244, 11.01509, 10.93011, 10.93055, 10.92047, ~
## $ lon       <dbl> 3.413116, 3.414022, 3.416367, 3.553716, 3.553230, 3.486705, ~
## $ year      <int> 2017, 2017, 2017, 2017, 2017, 2017, 2017, 2017, 2017, 2017, ~
## $ month     <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ day       <int> 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, ~
## $ date      <date> 2017-01-03, 2017-01-03, 2017-01-03, 2017-01-03, 2017-01-03,~
## $ ID        <int> 16993, 16894, 11728, 9560, 9574, 8938, 8914, 12410, 12422, 9~
## $ NDVI      <dbl> -0.035460993, 0.021276596, -0.027777778, -0.062500000, -0.05~
## $ NDBI      <dbl> 0.15527950, 0.10559006, 0.14110429, 0.20212766, 0.17441860, ~
## $ EVI       <dbl> -1.04166667, 1.07142857, -0.50000000, -0.36496350, -0.952380~
## $ landcover <chr> "builtup", "builtup", "builtup", "builtup", "builtup", "buil~
## $ veg       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, ~
## $ builtup   <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, ~
```

Vegetation and band values

Now we perform some data exploration to determine the relationship between vegetation and the band values. Also, we will look at the seasonality of the bands (i.e. which bands vary the most among landcover types).

Mean band values and vegetation

```
## Create a data set that is one row per location
## with mean(NDVI), mean(B7), and landcover type for each location
dm = d %>%
  group_by(ID) %>%
  summarise(
    B1   = mean(B1, na.rm=T),
    B2   = mean(B2, na.rm=T),
    B3   = mean(B3, na.rm=T),
    B4   = mean(B4, na.rm=T),
    B5   = mean(B5, na.rm=T),
    B6_VCID_1 = mean(B6_VCID_1, na.rm=T),
    B6_VCID_2 = mean(B6_VCID_2, na.rm=T),
    B7   = mean(B7, na.rm=T),
    NDVI = mean(NDVI, na.rm=T),
    NDBI = mean(NDBI, na.rm=T),
    EVI  = mean(EVI, na.rm=T),
    landcover = unique(landcover)) %>%
  ungroup() %>%
  mutate(veg = ifelse(landcover %in% c('natforest', 'orchard', 'cropland'),
                    1, 0),
         builtup = ifelse(landcover == 'builtup', 1, 0),
         EVI = ifelse(is.infinite(EVI), NA, EVI)) %>%
  as.data.frame()

## Inspect the resulting data frame
glimpse(dm)
```

```
## Rows: 400
## Columns: 15
## $ ID        <int> 2043, 2069, 2095, 2100, 2114, 2118, 2164, 2181, 2402, 2404, ~
## $ B1        <dbl> 80.43333, 83.48000, 90.95833, 87.91304, 85.92857, 85.92857, ~
## $ B2        <dbl> 69.16667, 73.52000, 81.70833, 78.30435, 77.42857, 77.42857, ~
## $ B3        <dbl> 73.43333, 78.92000, 95.04167, 87.82609, 91.89286, 91.89286, ~
## $ B4        <dbl> 91.36667, 94.56000, 97.08333, 95.30435, 75.10714, 75.10714, ~
## $ B5        <dbl> 79.00000, 87.12000, 111.33333, 102.04348, 93.03571, 93.03571~
## $ B6_VCID_1 <dbl> 130.1333, 131.3200, 130.7083, 131.4348, 139.0357, 139.0357, ~
## $ B6_VCID_2 <dbl> 147.7333, 150.0000, 148.8333, 149.8696, 164.2500, 164.2500, ~
## $ B7        <dbl> 51.30000, 57.92000, 79.20833, 71.13043, 76.96429, 76.96429, ~
## $ NDVI      <dbl> 0.150671964, 0.134296182, 0.047115152, 0.076682407, -0.09528~
## $ NDBI      <dbl> -0.087057210, -0.048786083, 0.054766177, 0.028438968, 0.1067~
## $ EVI       <dbl> -1.0751116, 1.1790665, -3.6038950, -0.2102671, -0.8309687, -~
## $ landcover <chr> "orchard", "orchard", "orchard", "orchard", "builtup", "buil~
## $ veg       <dbl> 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ builtup   <dbl> 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
```

We see there are 400 rows, one for each location, and that each row has `ID`, `landcover` type, and band values for the corresponding location.

Here is a summary of the mean band values for each landcover type.

```
dg = dm %>%
  select(-landcover, -builtup) %>%
  pivot_longer(cols = -c(ID, veg)) %>%
  group_by(name, veg) %>%
```

```r
  summarise(mean = mean(value,
                        na.rm = T)) %>%
  mutate(mean = round(mean, 2),
         veg = paste0('veg', veg)) %>%
  pivot_wider(names_from = veg,
              values_from = mean) %>%
  mutate(diff = veg1 - veg0)
```

```
## `summarise()` has grouped output by 'name'. You can override using the
## `.groups` argument.
```

```r
dg
```

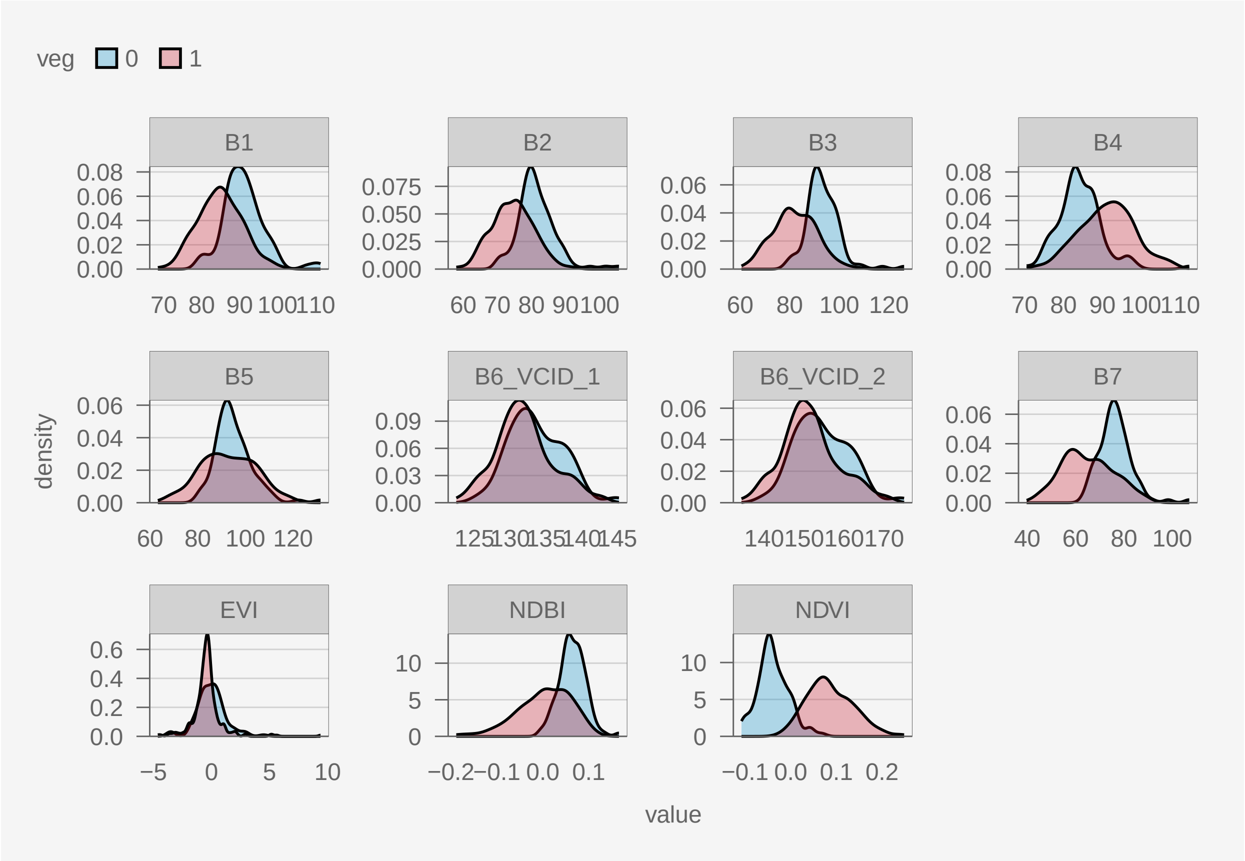
```
## # A tibble: 11 x 4
## # Groups:   name [11]
##    name        veg0   veg1   diff
##    <chr>      <dbl>  <dbl>  <dbl>
##  1 B1         91.0   84.8   -6.23
##  2 B2         81.5   74.5   -7.05
##  3 B3         93.4   82.0  -11.3
##  4 B4         84.4   91.2    6.88
##  5 B5         95.2   92.7   -2.42
##  6 B6_VCID_1 134.   132.    -1.70
##  7 B6_VCID_2 154.   151.    -2.98
##  8 B7         76.4   65    -11.4
##  9 EVI        -0.04  -0.39  -0.35
## 10 NDBI        0.06   0     -0.06
## 11 NDVI       -0.04   0.09   0.13
```

Histogram of all bands, separated by `veg`.

```r
dg = dm %>%
  select(-landcover, -builtup) %>%
  pivot_longer(cols = -c(ID, veg)) %>%
  mutate(veg = factor(veg))
head(dg)

g = ggplot(dg,
           aes(x = value,
               fill = veg)) +
  geom_density(alpha = 0.3) +
  facet_wrap(~name,
             scales = 'free')

g %>%
  pub(type = 'hist',
      facet = T,
      base_size = 9)
```




```
ggsave("img/density_plot_per_band.png", plot = g)
```

```
## Saving 6.5 x 4.5 in image

## # A tibble: 6 x 4
##      ID veg   name      value
##   <int> <fct> <chr>     <dbl>
## 1  2043 1     B1         80.4
## 2  2043 1     B2         69.2
## 3  2043 1     B3         73.4
## 4  2043 1     B4         91.4
## 5  2043 1     B5         79
## 6  2043 1     B6_VCID_1 130.
```

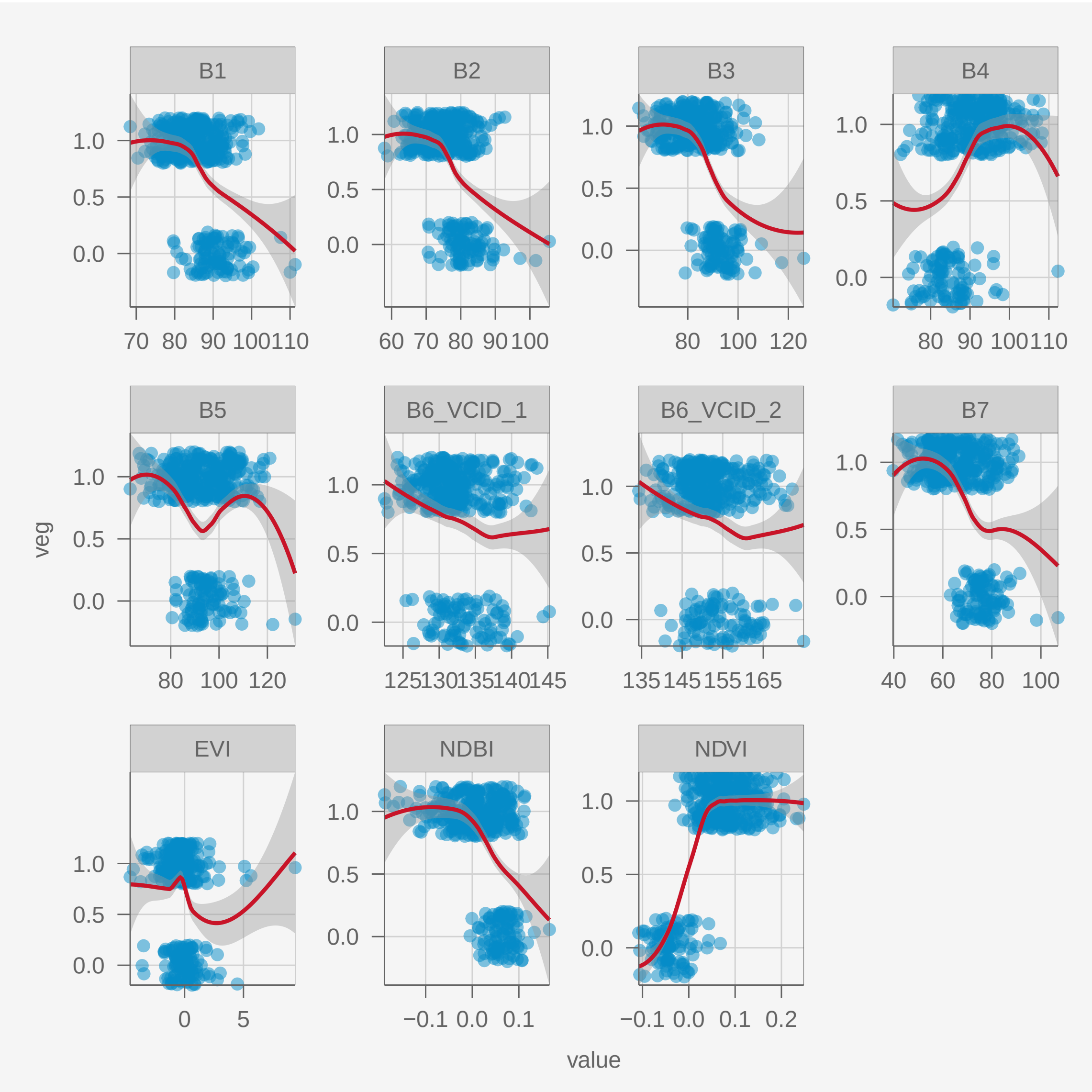
Scatter plot of bands vs veg

```
ds = dg %>%
  mutate(veg = as.numeric(as.character(veg)))

g = ggplot(ds,
           aes(x = value,
               y = veg)) +
  geom_jitter(alpha  = 0.5,
              height = 0.2,
              width  = 0,
              color=pubblue) +
  geom_smooth(color=pubred) +
  facet_wrap(~name,
```

```
            scales = 'free')

g %>%
  pub(type = 'scatter',
      facet = T,
      base_size = 9,
      ybreaks = c(0, 0.5, 1))
```




```
ggsave("img/scatter_plot_bands_vs_veg.png", plot = g)
```

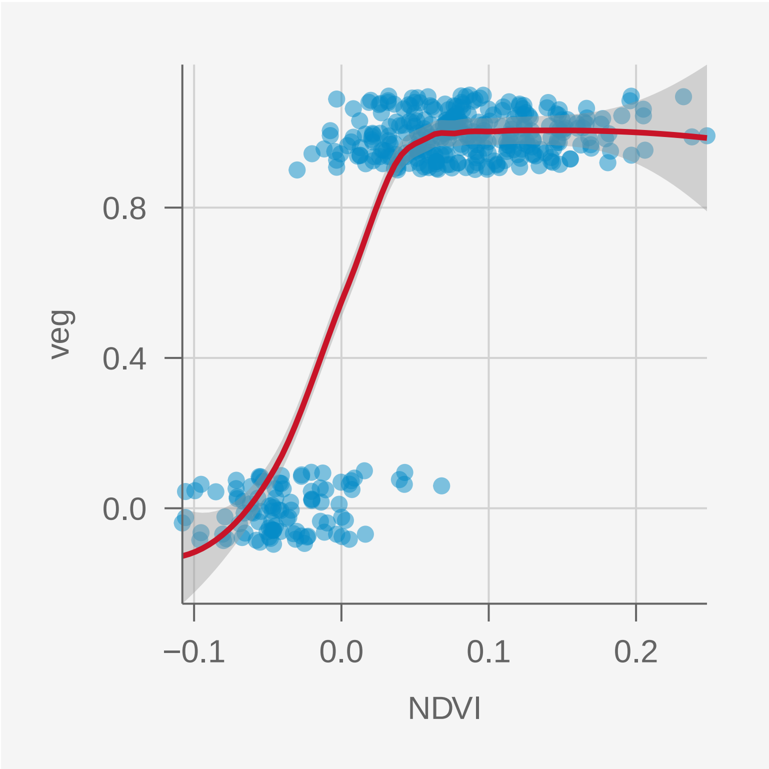
It seems most band values have a logistic relationship with `veg`.

Let's have a closer look at `NDVI` vs `veg`, since we know that `NDVI` is a good indicator of vegetation.

```
g = ggplot(dm,
       aes(NDVI,
           veg))+
  geom_jitter(height = 0.1,
```

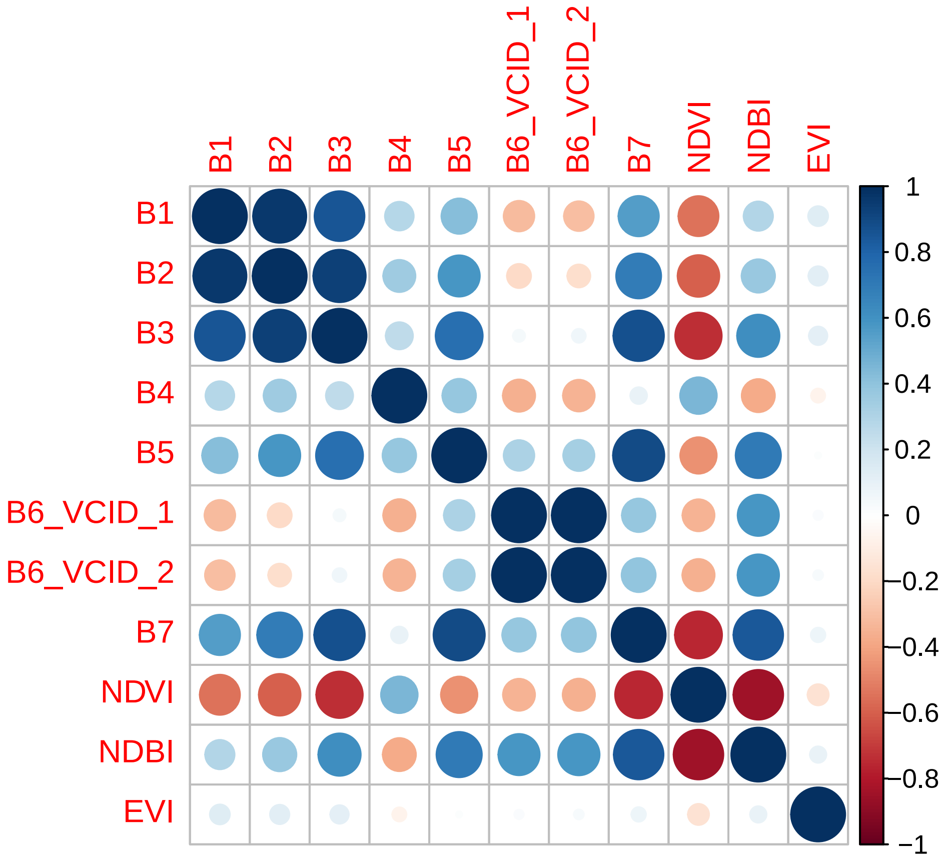
```
              width  = 0,
              alpha = 0.5,
              color = pubblue) +
  geom_smooth(color = pubred)

g %>% pub()
```



Corrplot of all bands

```
library(corrplot)
dcor = dm %>%
  select(-ID, -veg, -builtup, -landcover) %>%
  mutate(EVI = ifelse(is.infinite(EVI), NA, EVI)) %>%
  cor(use = 'pairwise.complete.obs')
dcor %>% round(2)
corrplot(dcor)
```

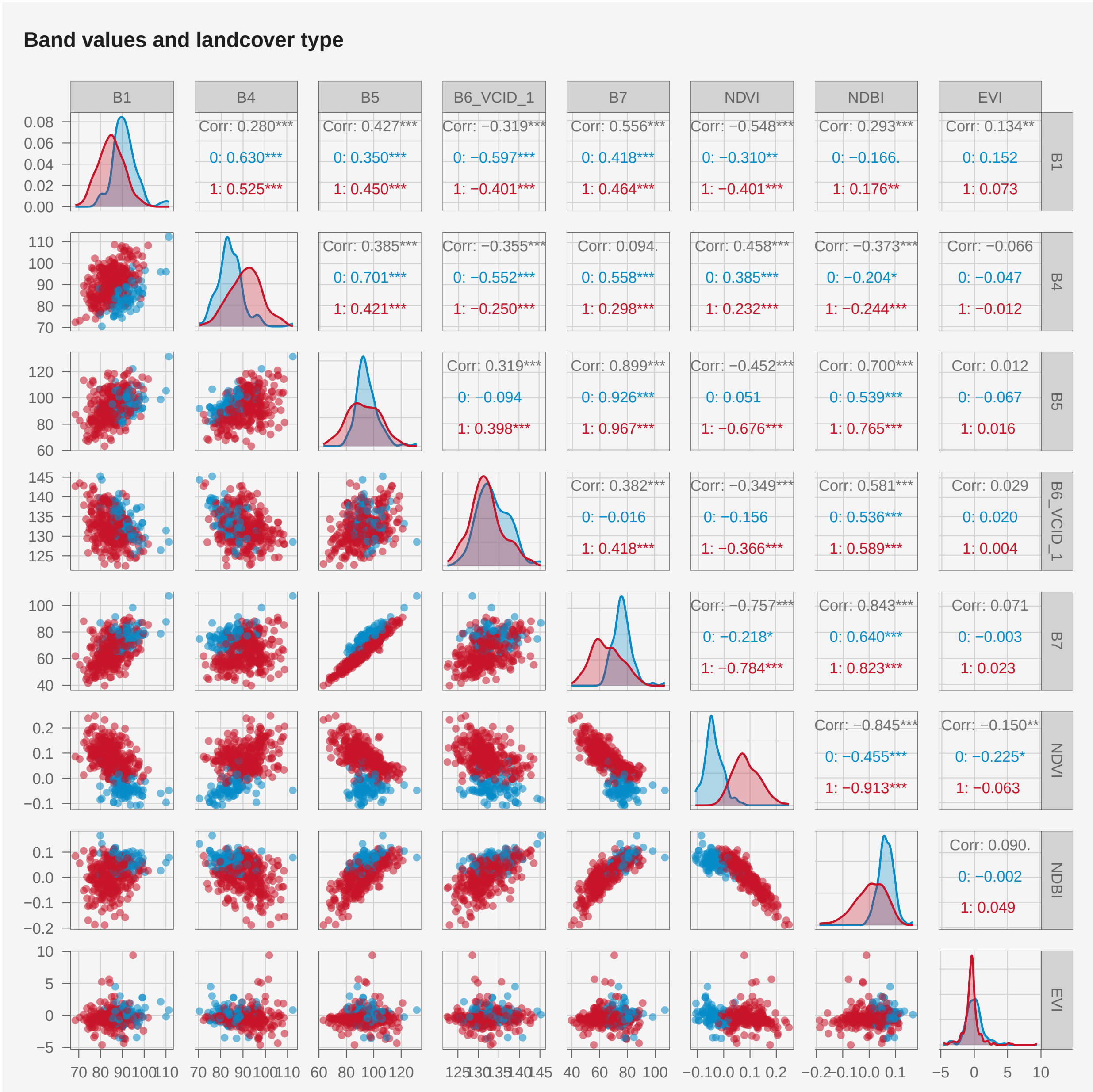



```
##                B1    B2    B3    B4    B5 B6_VCID_1 B6_VCID_2    B7  NDVI  NDBI
## B1           1.00  0.97  0.85  0.28  0.43     -0.32     -0.30  0.56 -0.55  0.29
## B2           0.97  1.00  0.94  0.35  0.58     -0.20     -0.18  0.69 -0.59  0.38
## B3           0.85  0.94  1.00  0.25  0.76      0.05      0.06  0.87 -0.74  0.61
## B4           0.28  0.35  0.25  1.00  0.38     -0.35     -0.34  0.09  0.46 -0.37
## B5           0.43  0.58  0.76  0.38  1.00      0.32      0.33  0.90 -0.45  0.70
## B6_VCID_1 -0.32 -0.20  0.05 -0.35  0.32      1.00      1.00  0.38 -0.35  0.58
## B6_VCID_2 -0.30 -0.18  0.06 -0.34  0.33      1.00      1.00  0.39 -0.35  0.59
## B7           0.56  0.69  0.87  0.09  0.90      0.38      0.39  1.00 -0.76  0.84
## NDVI        -0.55 -0.59 -0.74  0.46 -0.45     -0.35     -0.35 -0.76  1.00 -0.84
## NDBI         0.29  0.38  0.61 -0.37  0.70      0.58      0.59  0.84 -0.84  1.00
## EVI          0.13  0.13  0.12 -0.07  0.01      0.03      0.03  0.07 -0.15  0.09
##               EVI
## B1           0.13
## B2           0.13
## B3           0.12
## B4          -0.07
## B5           0.01
## B6_VCID_1    0.03
## B6_VCID_2    0.03
## B7           0.07
## NDVI        -0.15
## NDBI         0.09
## EVI          1.00
```

Pairs plot with points colored by `veg`. We omit some bands that are highly correlated with others to make the plot more readable.

```r
library(GGally)
title = 'Band values and landcover type'
dg = dm %>%
  mutate(veg = factor(veg))
bands = c('B1', #'B2', 'B3',
  'B4', 'B5',
  'B6_VCID_1', #'B6_VCID_2',
  'B7', 'NDVI', 'NDBI', 'EVI')

g = ggpairs(dg,
            aes(color = veg,
                fill  = veg,
                alpha = 0.1,
                shape = '20'),
            columns = bands,
            diag    = list(continuous = pub.density)) +
  labs(title    = title) +
  theme_pub(type = 'pairs',
            base_size = 8)
g
```




```r
ggsave("img/pair_plot_bands.png", plot = g)
```

We can tell from this plot which variables or pairs of variables will likely help. If we focus on the line `NDVI` = 0, we can see that as `NDBI` and `B7` increase the proportion of `veg` decreases. The "boundary line" between `veg = 1` and `veg = 0` looks like it has a negative slope, and there are more `veg = 1` above that boundary line.

These observations will be confirmed when fitting some models.

Seasonality

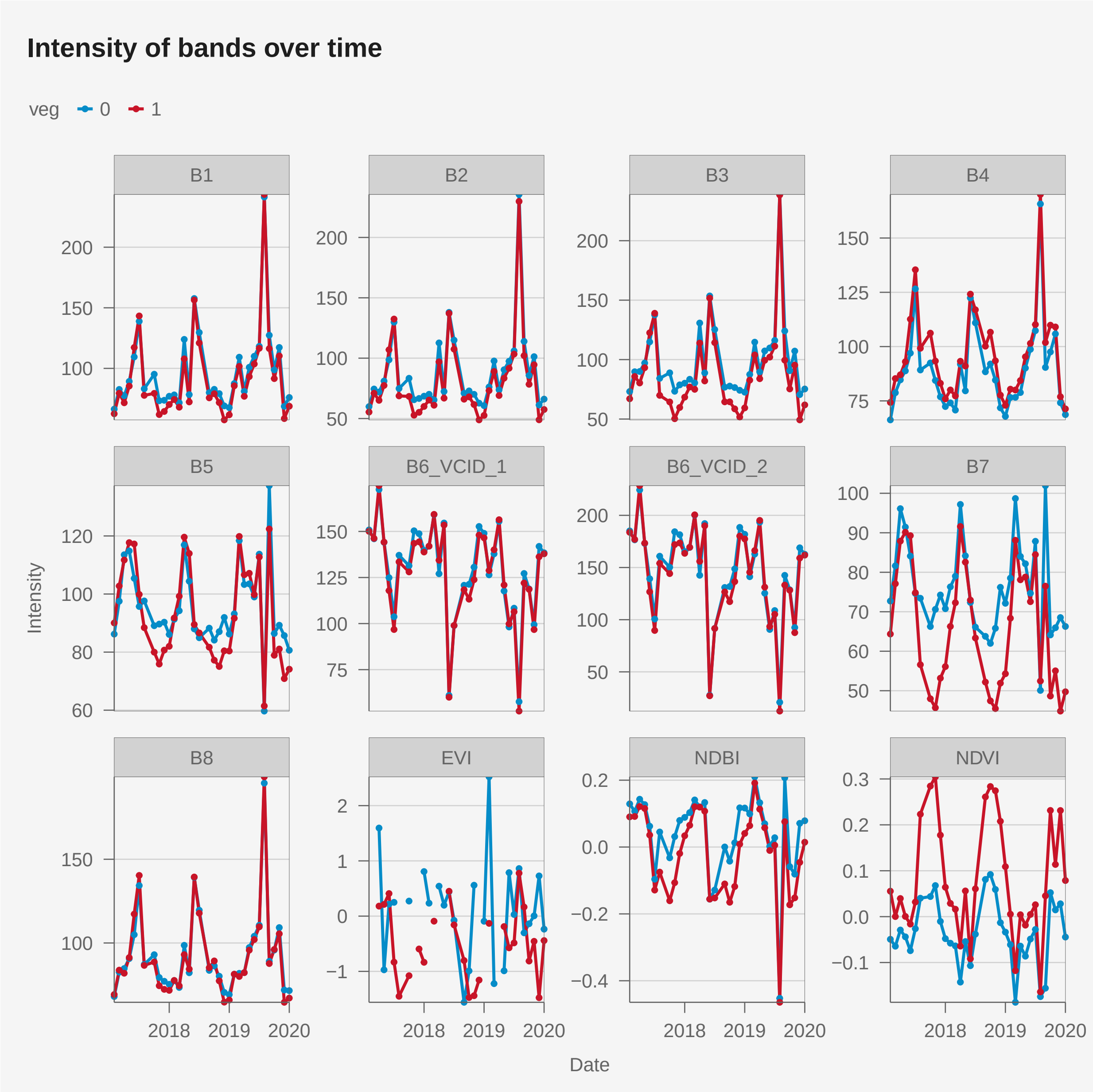
Bands over time for `veg = 1` and `veg = 0`.

```r
dd = d %>%
  filter(!is.infinite(EVI)) %>%
  select(-lat, -lon, -landcover, -builtup) %>%
  pivot_longer(cols = c(-ID, -veg, -year,
```

```
                        -month, -day, -date)) %>%
  mutate(year.mon = year+month/12) %>%
  group_by(veg,
           year.mon,
           name) %>%
  summarise(value = mean(value)) %>%
  mutate(veg = factor(veg))
head(dd)

title = "Intensity of bands over time"
g = ggplot(dd,
           aes(x = year.mon,
               y = value,
               color = veg))+
  geom_line(linewidth = .75)+
  geom_point(size = 1)+
  facet_wrap(~name,
             scales = 'free_y') +
  labs(title = title,
       x = 'Date',
       y = 'Intensity')

 g %>%
   pub(type = 'line',
       facet = T,
       base_size = 10,
       xbreaks = c(2018, 2019, 2020),
       xlabels = as.character(c(2018, 2019, 2020)))
```




```r
ggsave("img/line_plot_bands_over_time.png", plot = g)
```

```
## # A tibble: 6 x 4
## # Groups:   veg, year.mon [1]
##   veg   year.mon name      value
##   <fct>    <dbl> <chr>     <dbl>
## 1 0        2017. B1         66.4
## 2 0        2017. B2         59.9
## 3 0        2017. B3         73.1
## 4 0        2017. B4         66.3
## 5 0        2017. B5         86.2
## 6 0        2017. B6_VCID_1 151.
```

veg and mean NDVI

Observed proportion of 1s for different subsets of the predictor

Let's look at the proportion of `veg` for different subsets of `NDVI`.

We can bin our data using the function `cut_interval`.

```
dm = dm %>%
  mutate(bin = cut_interval(NDVI,
                            length = 0.05))
head(dd)
```

```
## # A tibble: 6 x 4
## # Groups:   veg, year.mon [1]
##   veg   year.mon name      value
##   <fct>    <dbl> <chr>     <dbl>
## 1 0        2017. B1         66.4
## 2 0        2017. B2         59.9
## 3 0        2017. B3         73.1
## 4 0        2017. B4         66.3
## 5 0        2017. B5         86.2
## 6 0        2017. B6_VCID_1 151.
```

Let's check the counts of `veg` in each bin.

```
bin.means = dm %>%
  group_by(bin, veg) %>%
  count() %>%
  pivot_wider(names_from = veg,
              values_from = n,
              values_fill = 0) %>%
  mutate(n = `0` + `1`,
         p = `1`/n)
bin.means
```

```
## # A tibble: 8 x 5
## # Groups:   bin [8]
##   bin              `0`   `1`     n     p
##   <fct>          <int> <int> <int> <dbl>
## 1 [-0.15,-0.1]       3     0     3 0
## 2 (-0.1,-0.05]      32     0    32 0
## 3 (-0.05,0]         51    10    61 0.164
## 4 (0,0.05]          13    63    76 0.829
## 5 (0.05,0.1]         1   122   123 0.992
## 6 (0.1,0.15]         0    72    72 1
## 7 (0.15,0.2]         0    27    27 1
## 8 (0.2,0.25]         0     6     6 1
```

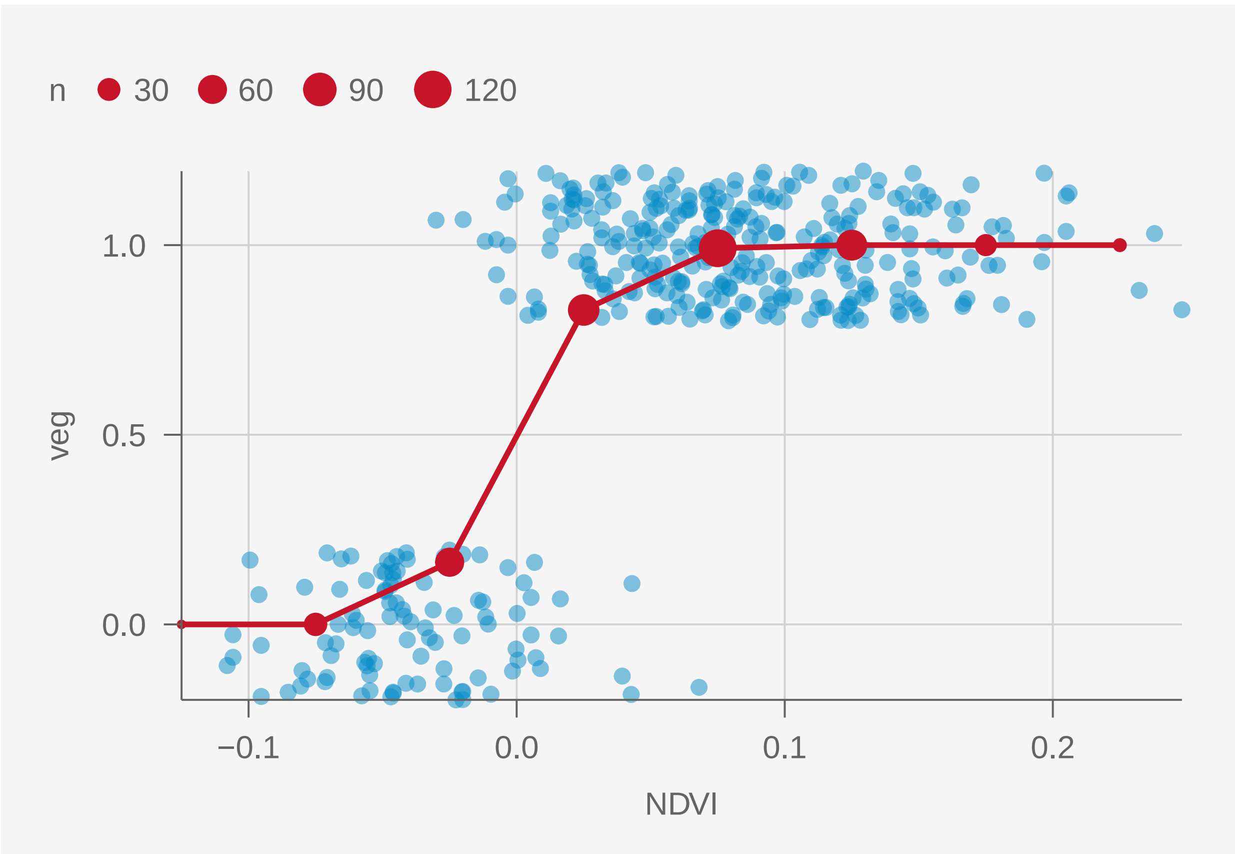
Let's plot these observed proportions on a scatter plot

```
bin.means = bin.means %>%
  ungroup() %>% #################### !!!!!!!!!!!!
  mutate(mid = seq(-.125, .225, by = 0.05))

g = ggplot(dm,
          aes(x = NDVI,
              y = veg)) +
```

```r
  geom_jitter(alpha = 0.5,
              height = 0.2,
              width = 0,
              color = pubblue) +
  geom_point(data = bin.means,
             aes(x = mid,
                 y = p,
                 size = n),
             color = pubred) +
  geom_line(data = bin.means,
            aes(x = mid,
                y = p),
            color = pubred)

g %>%
  pub(type = 'scatter',
      ybreaks = c(0, 0.5, 1))
```



Based on this plot, we can see that the proportion of `veg` is higher for higher values of `NDVI`. This is consistent with the fact that NDVI is a good indicator of vegetation. We also see that the relationship is not linear, but rather an S-shaped curve, so a logistic regression model might be a good choice for modeling this relationship.

Observed Proportion of 1s for different subsets of other predictors

Instead of looking at only `NDVI`, let's look at all of the band values and indexes. We could make a scatter plot for each statistic like we did for `NDVI` above, but it will be easier to reorganize the data and use `facet_wrap` as we have done before. So we'll we use the long format of the data `ds` from above.

```r
head(ds)
```

```
## # A tibble: 6 x 4
##      ID   veg name      value
##   <int> <dbl> <chr>     <dbl>
## 1  2043     1 B1         80.4
## 2  2043     1 B2         69.2
## 3  2043     1 B3         73.4
## 4  2043     1 B4         91.4
## 5  2043     1 B5         79
## 6  2043     1 B6_VCID_1 130.
```

We now find the proportion of `veg` for each bin of each band value. Here, we specify 10 bins for every band, since it would be hard to find an ideal number of bins per band.

```r
ds = ds %>%
  group_by(name) %>%
  mutate(bin = cut_interval(value, n=10))
head(ds)
```

```
## # A tibble: 6 x 5
## # Groups:   name [6]
##      ID   veg name      value bin
##   <int> <dbl> <chr>     <dbl> <fct>
## 1  2043     1 B1         80.4 (77,81.3]
## 2  2043     1 B2         69.2 (67.5,72.3]
## 3  2043     1 B3         73.4 (67.1,73.6]
## 4  2043     1 B4         91.4 (87.2,91.4]
## 5  2043     1 B5         79   (76.9,83.7]
## 6  2043     1 B6_VCID_1 130.  (129,132]
```

```r
dp = ds %>%
  group_by(name, bin) %>%
  summarise(p = mean(veg),
            n = n())
```

```
## `summarise()` has grouped output by 'name'. You can override using the
## `.groups` argument.
```

```r
head(dp)
```

```
## # A tibble: 6 x 4
## # Groups:   name [1]
##   name  bin             p     n
##   <chr> <fct>       <dbl> <int>
## 1 B1    [68.4,72.7] 1         4
## 2 B1    (72.7,77]   1        27
## 3 B1    (77,81.3]   0.929    56
## 4 B1    (81.3,85.6] 0.946    92
## 5 B1    (85.6,89.9] 0.673   104
## 6 B1    (89.9,94.2] 0.557    79
```

Let's find the midpoint of each interval, since we'll need that for plotting. Since we don't want to write down 11 different formulas using `seq` like we did above for `NDVI` to find the midpoints, we'll extract the left and right coordinates from the bin using regular expressions, and use those to compute the midpoint.

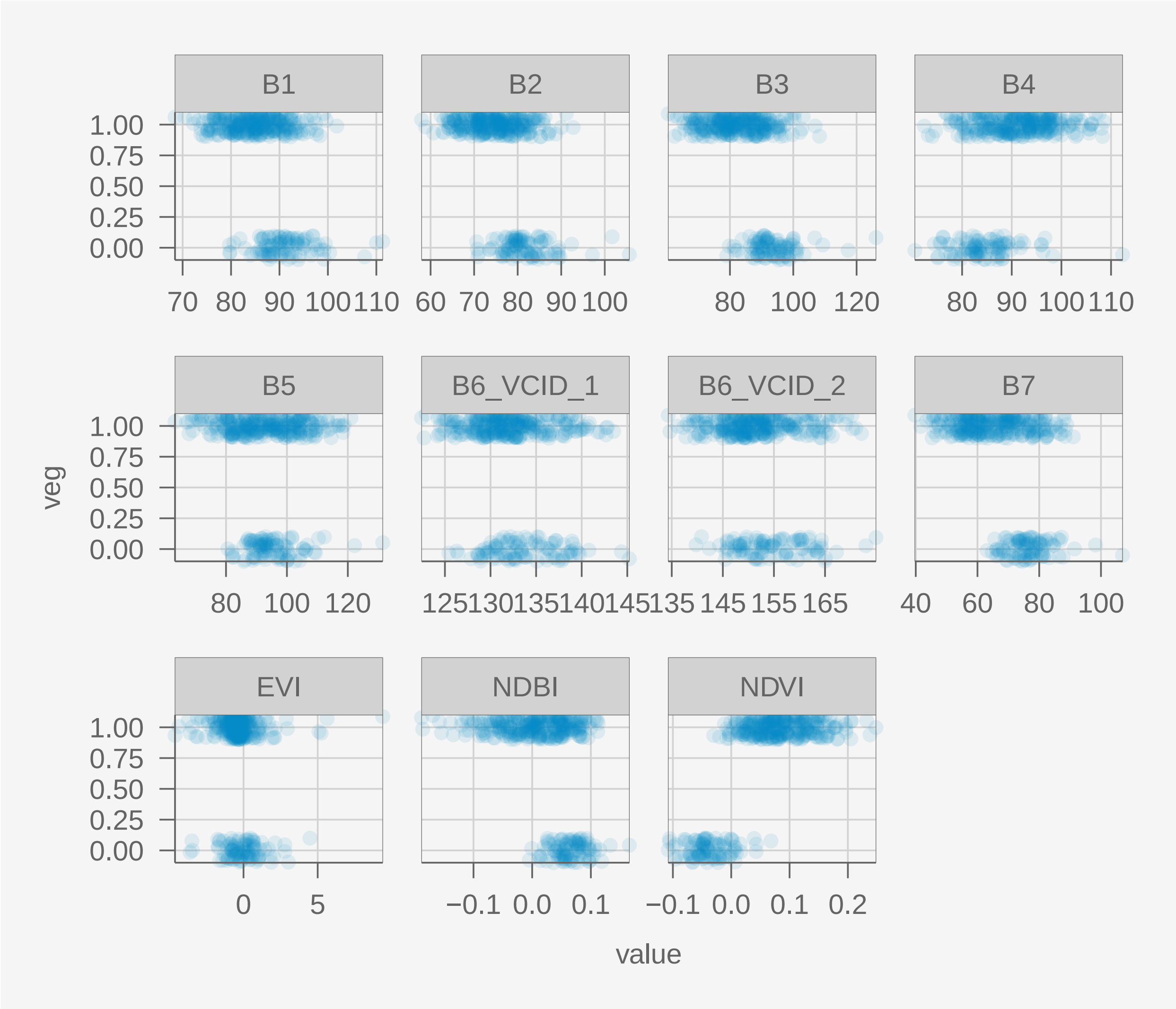
```r
dp = dp %>%
  mutate(left  = gsub(',.+', '', bin),
         left  = gsub('[(]|[[]', '', left),
         right = gsub('.+,|[]]', '', bin),
         left  = as.numeric(left),
         right = as.numeric(right),
         mid   = left/2 + right/2)
head(dp)
```

```
## # A tibble: 6 x 7
## # Groups:   name [1]
##   name  bin             p     n  left right   mid
##   <chr> <fct>       <dbl> <int> <dbl> <dbl> <dbl>
## 1 B1    [68.4,72.7] 1         4  68.4  72.7  70.6
## 2 B1    (72.7,77]   1        27  72.7  77    74.8
## 3 B1    (77,81.3]   0.929    56  77    81.3  79.2
## 4 B1    (81.3,85.6] 0.946    92  81.3  85.6  83.4
## 5 B1    (85.6,89.9] 0.673   104  85.6  89.9  87.8
## 6 B1    (89.9,94.2] 0.557    79  89.9  94.2  92.1
```

Let's now make a plot with the value of the stat on the horizontal axis, `veg` on the vertical axis, and let's use `facet_wrap` to make a different window for each stat.

We'll start with just the scatter plots.

```r
g = ggplot(ds,
           aes(x = value,
               y = veg))+
  geom_jitter(height = 0.1,
              width  = 0,
              alpha  = 0.1,
              color = pubblue)+
  facet_wrap(~name,
             scales = 'free_x')
g %>%
  pub(facet = T)
```



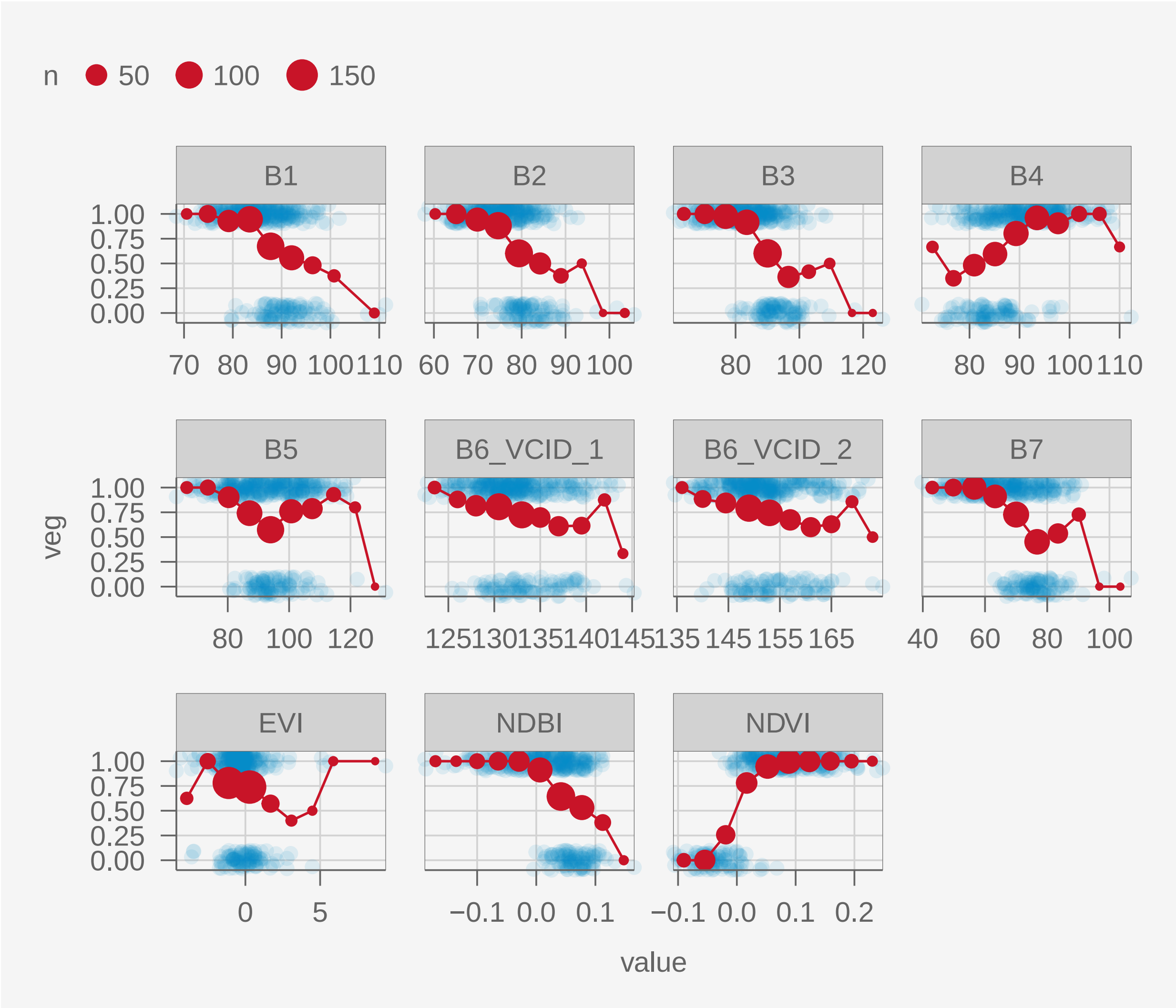
Now let's add proportions.

```
g2 = g +
  geom_point(data = dp,
             aes(x    = mid,
                 y    = p,
                 size = n),
             color = pubred) +

  geom_line(data = dp,
            aes(x = mid,
                y = p),
            color = pubred,
            linewidth = .5) +

  scale_size(range=c(0.5, 3))

g2 %>%
  pub(facet = T)
```




```
ggsave("img/scatter_plot_bands_vs_veg_dot.png", plot = g2)
```

As we would expect, there is a positive relationship between `NDVI` and `veg` but a negative relationship between `NDBI` and `veg`. We can get a rough idea of the relative strength of these relationships as well by looking at the steepness of the curve. For example, `NDVI` appears to have a very strong relationship to `B4` is less strong.

We also see that the S-shaped curve appears a lot, especially if you ignore the left and right extremes where there are few data points. We have sized the red dots using the number of observations in each subset, so the small red dots correspond to the subsets with very few data points.