SRFN project analysis

Marissa Dyck

2025-1-10

Contents

The first two chunks of this r markdown file after the r setup allow for plot zooming, but it also means that the html file must be opened in a browser to view the document properly. When it knits in RStudio the preview will appear empty but the html when opened in a browser will have all the info and you can click on each plot to Zoom in on it.

Before you begin

Notes

A few notes about this script.

If you are running this make sure you download the whole SRFN (GitHub repository)[https://github.com/marissadyck/SRFN_ACME_Camera_Project] from my GitHub. This will ensure you have all the files, data, and proper folder structure you will need to run this code and associated analyses.

Also make sure you open RStudio through the R project (ARFN_ACME_Camera_Project.Rproj) this will automatically set your working directory to the correct place (wherever you saved the repository and it's files) and ensure you don't have to change the file paths for some of the data.

Lastly, if you are looking to adapt this code for a future year of data, you will want to ensure you have run all the code through 2_ACME_SRFN_landscape_covariate_exploration_script.Rmd with your data as there is much data formatting, cleaning, and restructuring that has to be done before this code will work. *Helpful note: The files are numbered in the order they are used to prep for this analysis.*

If you have question please email the most recent author, currently

Marissa A. Dyck
Postdoctoral research fellow
University of Victoria
School of Environmental Studies
Email: marissadyck17@gmail.com

R and RStudio

Before starting you should ensure you have the latest version of R and RStudio downloaded. This code was generated under R version 4.2.3 and with RStudio version 2024.04.2+764.

You can download R and RStudio HERE

R markdown

This script is written in R markdown and thus uses a mix of coding markup languages and R. If you are planning to run this script with new data or make any modifications you will want to be familiar with some basics of R markdown.

Below is an R markdown cheatsheet to help you get started,
R markdown cheatsheet

Install packages

If you don't already have the following packages installed, use the code below to install them. *NOTE this will not run automatically as eval=FALSE is included in the chunk setup (i.e. I don't want it to run every time I run this code since I have the packages installed).

```
install.packages('tidyverse')
install.packages('ggpubr')
install.packages('corrplot')
install.packages('Hmisc')
install.packages('glmmTMB')
```

```
install.packages('MuMIn')
install.packages('TMB', type = 'source')
install.packages('rphylopic')
install.packages('broom')
install.packages('lme4')
install.packages('car')
```

Load libraries

Then load the packages to your library so they are usable for this session.

```
library(tidyverse) # data tidying, visualization, and much more; this will load all tidyverse packages,
library(ggpubr) # make modifications to plot for publication (arrange plots)
library(PerformanceAnalytics)     # Used to generate a correlation plot
library(Hmisc) # used to generate histograms for all variables in data frame
library(glmmTMB)      # Constructing GLMMs
```

```
## Warning in checkMatrixPackageVersion(): Package version inconsistency detected.
## TMB was built with Matrix version 1.4.1
## Current Matrix version is 1.5.3
## Please re-install 'TMB' from source using install.packages('TMB', type = 'source') or ask CRAN for a
```

```
## Warning in checkDepPackageVersion(dep_pkg = "TMB"): Package version inconsistency detected.
## glmmTMB was built with TMB version 1.9.6
## Current TMB version is 1.9.1
## Please re-install glmmTMB from source or restore original 'TMB' package (see '?reinstalling' for mor
```

```
library(MuMIn) # for model selection
library(rphylopic) # add animal silhouettes to graphs
library(broom) # extracting odds ratios in a tidy format
library(lme4) # contructing generalized linear mixed effects models
library(car) # used for calculating variance inflation factor (VIF) to assess model fit
```

Analysis prep

Now we can start the analysis prep.

First we need to read in the proportional detection (response metrics) and covariate (explanatory metrics) data files for all 6 LUs (fiscal years 2021-2022 and 2022-2023)

Response metrics

We have multiple response metric files that were generated in script #1, since there were a few very rare species we want to look at, let's load in both files for now and make sure they are formatted properly for the analysis

Import response metric data

I'm going to load them all at once using purrr and we can separate them later depending on what we want to use them for

```
response_metrics <- file.path('data/processed',

                              # provide file names
                             c('srfn_proportional_detections.csv',
                              'srfn_total_detections.csv',
                              'srfn_presence_absence.csv')) %>%

  # use purrr map to read in all files
  map(~.x %>%

        # use tidyverse read_csv
        read_csv(.,

                 # specify how some columns are read in
                 col_types = cols(site = col_factor())) %>%

        # set column names to lowercase for coding ease
        set_names(
          names(.) %>%
            tolower())) %>%

  # set names for list items
   purrr::set_names('prop_detections',
                    'total_detections',
                    'presence_absence')
```

Data checks

```
str(response_metrics)
```

```
## List of 3
##  $ prop_detections : spc_tbl_ [63 x 29] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##   ..$ site                   : Factor w/ 63 levels "LUAG_119","LUAG_124",..: 1 2 3 4 5 6 7 8 9 10 .
##   ..$ black_bear             : num [1:63] 4 1 3 1 0 0 6 3 0 4 ...
##   ..$ coyote                 : num [1:63] 8 2 6 2 0 1 11 7 1 8 ...
##   ..$ red_fox                : num [1:63] 1 1 1 1 0 0 1 2 0 0 ...
##   ..$ white-tailed_deer      : num [1:63] 13 12 9 16 11 7 14 8 5 11 ...
##   ..$ fisher                 : num [1:63] 0 1 0 0 0 0 0 1 0 0 ...
##   ..$ grey_wolf              : num [1:63] 0 1 0 1 0 0 2 3 0 3 ...
##   ..$ grizzly_bear           : num [1:63] 0 1 0 0 0 0 0 0 0 0 ...
##   ..$ marten                 : num [1:63] 0 1 2 0 0 0 0 0 0 0 ...
##   ..$ snowshoe_hare          : num [1:63] 0 5 1 0 1 5 1 3 0 0 ...
##   ..$ elk                    : num [1:63] 0 0 7 2 0 0 0 0 0 8 ...
##   ..$ moose                  : num [1:63] 0 0 4 8 11 1 7 7 13 3 ...
##   ..$ mule_deer              : num [1:63] 0 0 1 3 0 1 0 0 0 0 ...
##   ..$ lynx                   : num [1:63] 0 0 0 0 0 1 1 2 0 0 ...
```

```
##   ..$ cougar                  : num [1:63] 0 0 0 0 0 0 1 0 0 0 ...
##   ..$ absent_black_bear       : num [1:63] 9 12 6 15 16 14 8 10 14 8 ...
##   ..$ absent_coyote           : num [1:63] 8 13 5 17 19 17 6 9 17 7 ...
##   ..$ absent_red_fox          : num [1:63] 15 14 10 18 19 18 16 14 18 15 ...
##   ..$ absent_white-tailed_deer: num [1:63] 3 3 2 3 8 11 3 8 13 4 ...
##   ..$ absent_fisher           : num [1:63] 16 14 11 19 19 18 17 15 18 15 ...
##   ..$ absent_grey_wolf        : num [1:63] 16 14 11 18 19 18 15 13 18 12 ...
##   ..$ absent_grizzly_bear     : num [1:63] 13 12 9 16 16 14 14 13 14 12 ...
##   ..$ absent_marten           : num [1:63] 16 14 9 19 19 18 17 16 18 15 ...
##   ..$ absent_snowshoe_hare    : num [1:63] 16 10 10 19 18 13 16 13 18 15 ...
##   ..$ absent_elk              : num [1:63] 16 15 4 17 19 18 17 16 18 7 ...
##   ..$ absent_moose            : num [1:63] 16 15 7 11 8 17 10 9 5 12 ...
##   ..$ absent_mule_deer        : num [1:63] 16 15 10 16 19 17 17 16 18 15 ...
##   ..$ absent_lynx             : num [1:63] 16 15 11 19 19 17 16 14 18 15 ...
##   ..$ absent_cougar           : num [1:63] 16 15 11 19 19 18 16 16 18 15 ...
##   ..- attr(*, "spec")=
##   .. .. cols(
##   .. ..   site = col_factor(levels = NULL, ordered = FALSE, include_na = FALSE),
##   .. ..   black_bear = col_double(),
##   .. ..   coyote = col_double(),
##   .. ..   red_fox = col_double(),
##   .. ..   `white-tailed_deer` = col_double(),
##   .. ..   fisher = col_double(),
##   .. ..   grey_wolf = col_double(),
##   .. ..   grizzly_bear = col_double(),
##   .. ..   marten = col_double(),
##   .. ..   snowshoe_hare = col_double(),
##   .. ..   elk = col_double(),
##   .. ..   moose = col_double(),
##   .. ..   mule_deer = col_double(),
##   .. ..   lynx = col_double(),
##   .. ..   cougar = col_double(),
##   .. ..   absent_black_bear = col_double(),
##   .. ..   absent_coyote = col_double(),
##   .. ..   absent_red_fox = col_double(),
##   .. ..   `absent_white-tailed_deer` = col_double(),
##   .. ..   absent_fisher = col_double(),
##   .. ..   absent_grey_wolf = col_double(),
##   .. ..   absent_grizzly_bear = col_double(),
##   .. ..   absent_marten = col_double(),
##   .. ..   absent_snowshoe_hare = col_double(),
##   .. ..   absent_elk = col_double(),
##   .. ..   absent_moose = col_double(),
##   .. ..   absent_mule_deer = col_double(),
##   .. ..   absent_lynx = col_double(),
##   .. ..   absent_cougar = col_double()
##   .. .. )
##   ..- attr(*, "problems")=<externalptr>
##  $ total_detections: spc_tbl_ [63 x 33] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##   ..$ array            : chr [1:63] "LUAG" "LUAG" "LUAG" "LUBF" ...
##   ..$ site_number      : num [1:63] 119 124 137 104 105 12 121 126 130 132 ...
##   ..$ site             : Factor w/ 63 levels "LUAG_119","LUAG_124",..: 1 2 3 4 5 6 7 8 9 10 ...
##   ..$ black_bear       : num [1:63] 4 2 17 1 0 0 18 5 0 12 ...
##   ..$ coyote           : num [1:63] 21 3 23 2 0 1 28 21 2 27 ...
```

```
##   ..$ red_fox          : num [1:63] 4 1 2 1 0 0 1 2 0 0 ...
##   ..$ white-tailed_deer: num [1:63] 62 141 153 48 22 32 187 187 10 666 ...
##   ..$ domestic_dog     : num [1:63] 0 1 1 0 0 0 0 0 0 0 ...
##   ..$ fisher           : num [1:63] 0 1 0 0 0 0 0 1 0 0 ...
##   ..$ grey_wolf        : num [1:63] 0 2 0 1 0 0 4 6 0 26 ...
##   ..$ grizzly_bear     : num [1:63] 0 1 0 0 0 0 0 0 0 0 ...
##   ..$ marten           : num [1:63] 0 1 3 0 0 0 0 0 0 0 ...
##   ..$ snowshoe_hare    : num [1:63] 0 20 1 0 1 21 2 19 0 0 ...
##   ..$ elk              : num [1:63] 0 0 43 2 0 0 0 0 0 66 ...
##   ..$ moose            : num [1:63] 0 0 4 9 20 1 16 22 148 3 ...
##   ..$ mule_deer        : num [1:63] 0 0 4 8 0 1 0 0 0 0 ...
##   ..$ unknown          : num [1:63] 0 0 2 3 0 9 1 0 12 1 ...
##   ..$ unknown_ungulate : num [1:63] 0 0 41 4 0 0 0 0 7 0 ...
##   ..$ lynx             : num [1:63] 0 0 0 0 0 1 1 3 0 0 ...
##   ..$ cougar           : num [1:63] 0 0 0 0 0 0 1 0 0 0 ...
##   ..$ red_squirrel     : num [1:63] 0 0 0 0 0 0 0 1 0 0 ...
##   ..$ other            : num [1:63] 0 0 0 0 0 0 0 0 14 0 ...
##   ..$ atver            : num [1:63] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ unknown_deer     : num [1:63] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ unknown_bear     : num [1:63] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ other_birds      : num [1:63] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ human            : num [1:63] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ staff            : num [1:63] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ spruce_grouse    : num [1:63] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ hunter           : num [1:63] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ ruffed_grouse    : num [1:63] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ raven            : num [1:63] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ owl              : num [1:63] 0 0 0 0 0 0 0 0 0 0 ...
##   ..- attr(*, "spec")=
##   .. .. cols(
##   .. ..   array = col_character(),
##   .. ..   site_number = col_double(),
##   .. ..   site = col_factor(levels = NULL, ordered = FALSE, include_na = FALSE),
##   .. ..   black_bear = col_double(),
##   .. ..   coyote = col_double(),
##   .. ..   red_fox = col_double(),
##   .. ..   `white-tailed_deer` = col_double(),
##   .. ..   domestic_dog = col_double(),
##   .. ..   fisher = col_double(),
##   .. ..   grey_wolf = col_double(),
##   .. ..   grizzly_bear = col_double(),
##   .. ..   marten = col_double(),
##   .. ..   snowshoe_hare = col_double(),
##   .. ..   elk = col_double(),
##   .. ..   moose = col_double(),
##   .. ..   mule_deer = col_double(),
##   .. ..   unknown = col_double(),
##   .. ..   unknown_ungulate = col_double(),
##   .. ..   lynx = col_double(),
##   .. ..   cougar = col_double(),
##   .. ..   red_squirrel = col_double(),
##   .. ..   other = col_double(),
##   .. ..   atver = col_double(),
##   .. ..   unknown_deer = col_double(),
```

```
##   .. ..   unknown_bear = col_double(),
##   .. ..   other_birds = col_double(),
##   .. ..   human = col_double(),
##   .. ..   staff = col_double(),
##   .. ..   spruce_grouse = col_double(),
##   .. ..   hunter = col_double(),
##   .. ..   ruffed_grouse = col_double(),
##   .. ..   raven = col_double(),
##   .. ..   owl = col_double()
##   .. .. )
##   ..- attr(*, "problems")=<externalptr>
##  $ presence_absence: spc_tbl_ [63 x 33] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##   ..$ array            : chr [1:63] "LUAG" "LUAG" "LUAG" "LUBF" ...
##   ..$ site_number      : num [1:63] 119 124 137 104 105 12 121 126 130 132 ...
##   ..$ site             : Factor w/ 63 levels "LUAG_119","LUAG_124",..: 1 2 3 4 5 6 7 8 9 10 ...
##   ..$ black_bear       : num [1:63] 1 1 1 1 0 0 1 1 0 1 ...
##   ..$ coyote           : num [1:63] 1 1 1 1 0 1 1 1 1 1 ...
##   ..$ red_fox          : num [1:63] 1 1 1 1 0 0 1 1 0 0 ...
##   ..$ white-tailed_deer: num [1:63] 1 1 1 1 1 1 1 1 1 1 ...
##   ..$ domestic_dog     : num [1:63] 0 1 1 0 0 0 0 0 0 0 ...
##   ..$ fisher           : num [1:63] 0 1 0 0 0 0 0 1 0 0 ...
##   ..$ grey_wolf        : num [1:63] 0 1 0 1 0 0 1 1 0 1 ...
##   ..$ grizzly_bear     : num [1:63] 0 1 0 0 0 0 0 0 0 0 ...
##   ..$ marten           : num [1:63] 0 1 1 0 0 0 0 0 0 0 ...
##   ..$ snowshoe_hare    : num [1:63] 0 1 1 0 1 1 1 1 0 0 ...
##   ..$ elk              : num [1:63] 0 0 1 1 0 0 0 0 0 1 ...
##   ..$ moose            : num [1:63] 0 0 1 1 1 1 1 1 1 1 ...
##   ..$ mule_deer        : num [1:63] 0 0 1 1 0 1 0 0 0 0 ...
##   ..$ unknown          : num [1:63] 0 0 1 1 0 1 1 0 1 1 ...
##   ..$ unknown_ungulate : num [1:63] 0 0 1 1 0 0 0 0 1 0 ...
##   ..$ lynx             : num [1:63] 0 0 0 0 0 1 1 1 0 0 ...
##   ..$ cougar           : num [1:63] 0 0 0 0 0 0 1 0 0 0 ...
##   ..$ red_squirrel     : num [1:63] 0 0 0 0 0 0 0 1 0 0 ...
##   ..$ other            : num [1:63] 0 0 0 0 0 0 0 0 1 0 ...
##   ..$ atver            : num [1:63] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ unknown_deer     : num [1:63] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ unknown_bear     : num [1:63] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ other_birds      : num [1:63] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ human            : num [1:63] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ staff            : num [1:63] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ spruce_grouse    : num [1:63] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ hunter           : num [1:63] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ ruffed_grouse    : num [1:63] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ raven            : num [1:63] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ owl              : num [1:63] 0 0 0 0 0 0 0 0 0 0 ...
##   ..- attr(*, "spec")=
##   .. .. cols(
##   .. ..   array = col_character(),
##   .. ..   site_number = col_double(),
##   .. ..   site = col_factor(levels = NULL, ordered = FALSE, include_na = FALSE),
##   .. ..   black_bear = col_double(),
##   .. ..   coyote = col_double(),
##   .. ..   red_fox = col_double(),
##   .. ..   `white-tailed_deer` = col_double(),
```

```
##   .. ..   domestic_dog = col_double(),
##   .. ..   fisher = col_double(),
##   .. ..   grey_wolf = col_double(),
##   .. ..   grizzly_bear = col_double(),
##   .. ..   marten = col_double(),
##   .. ..   snowshoe_hare = col_double(),
##   .. ..   elk = col_double(),
##   .. ..   moose = col_double(),
##   .. ..   mule_deer = col_double(),
##   .. ..   unknown = col_double(),
##   .. ..   unknown_ungulate = col_double(),
##   .. ..   lynx = col_double(),
##   .. ..   cougar = col_double(),
##   .. ..   red_squirrel = col_double(),
##   .. ..   other = col_double(),
##   .. ..   atver = col_double(),
##   .. ..   unknown_deer = col_double(),
##   .. ..   unknown_bear = col_double(),
##   .. ..   other_birds = col_double(),
##   .. ..   human = col_double(),
##   .. ..   staff = col_double(),
##   .. ..   spruce_grouse = col_double(),
##   .. ..   hunter = col_double(),
##   .. ..   ruffed_grouse = col_double(),
##   .. ..   raven = col_double(),
##   .. ..   owl = col_double()
##   .. .. )
##   ..- attr(*, "problems")=<externalptr>
```

First checks look good, but let's remove a few species we aren't interested in modeling for this analysis from all three data sets

```
# first create focal species object with only species we have enough data to model in one response metr
glm_focal_vars <- c('site',
                    'white-tailed_deer',
                    'black_bear',
                    'coyote',
                    'elk',
                    'grey_wolf',
                    'grizzly_bear',
                    'lynx',
                    'moose',
                    'mule_deer',
                    'red_fox',
                    'snowshoe_hare')


response_metrics_subset <- response_metrics %>%


  map(~.x %>%

        # use purrr map to select only columns that match the focal species in all data sets
        select(matches(paste(glm_focal_vars,
```

```
                              collapse = '|')))
      )
```

Now we should subset each data frame individually even further

We prioritize the proportional monthly detection data as it gives us the most information, so for any species we have enough (we think) data for we will keep them in this data frame, otherwise we will remove

```
response_metrics_subset$prop_detections <- response_metrics_subset$prop_detections %>%

  # remove mule deer, red fox, and grizzly bear
  select(-contains(c('red_fox',
                     'mule_deer',
                     'grizzly_bear')))

head(response_metrics_subset$prop_detections)
```

```
## # A tibble: 6 x 17
##   site     black_bear coyote `white-tailed_deer` grey_wolf snowshoe_hare   elk
##   <fct>         <dbl>  <dbl>               <dbl>     <dbl>          <dbl> <dbl>
## 1 LUAG_119          4      8                  13         0              0     0
## 2 LUAG_124          1      2                  12         1              5     0
## 3 LUAG_137          3      6                   9         0              1     7
## 4 LUBF_104          1      2                  16         1              0     2
## 5 LUBF_105          0      0                  11         0              1     0
## 6 LUBF_12           0      1                   7         0              5     0
## # i 10 more variables: moose <dbl>, lynx <dbl>, absent_black_bear <dbl>,
## #   absent_coyote <dbl>, `absent_white-tailed_deer` <dbl>,
## #   absent_grey_wolf <dbl>, absent_snowshoe_hare <dbl>, absent_elk <dbl>,
## #   absent_moose <dbl>, absent_lynx <dbl>
```

Now moving onto our alternative response metrics, I'm not sure which we will use yet for the remaining three species so we will keep all three in both

Basically this code will be the inverse of the code above to simply keep only those species instead of remove them

```
# presence absence
response_metrics_subset$presence_absence <- response_metrics_subset$presence_absence %>%

  # remove mule deer, red fox, and grizzly bear
  select(contains(c('site',
                    'red_fox',
                     'mule_deer',
                     'grizzly_bear')))

head(response_metrics_subset$presence_absence)
```

```
## # A tibble: 6 x 5
##   site_number site     red_fox mule_deer grizzly_bear
##         <dbl> <fct>      <dbl>     <dbl>        <dbl>
## 1         119 LUAG_119       1         0            0
## 2         124 LUAG_124       1         0            1
```

```
## 3         137 LUAG_137       1         1            0
## 4         104 LUBF_104       1         1            0
## 5         105 LUBF_105       0         0            0
## 6          12 LUBF_12        0         1            0
```

```
# total detections
response_metrics_subset$total_detections <- response_metrics_subset$total_detections %>%

  # remove mule deer, red fox, and grizzly bear
  select(contains(c('site',
                    'red_fox',
                     'mule_deer',
                     'grizzly_bear')))

head(response_metrics_subset$total_detections)
```

```
## # A tibble: 6 x 5
##   site_number site     red_fox mule_deer grizzly_bear
##         <dbl> <fct>      <dbl>     <dbl>        <dbl>
## 1         119 LUAG_119       4         0            0
## 2         124 LUAG_124       1         0            1
## 3         137 LUAG_137       2         4            0
## 4         104 LUBF_104       1         8            0
## 5         105 LUBF_105       0         0            0
## 6          12 LUBF_12        0         1            0
```

Okay now that these are cleaned up we can remove the full response metrics list from our environment so we don't use it

```
rm(response_metrics)
```

Covariates

We also need our potential explanatory variables

Read in data

In the previous script, 2_ACME_SRFN_landscape_covariate_exploration_script.Rmd we cleaned up the covariates data let's also read that in as we will need to join it with the response metric data to run the models

```
covariates <- read_csv('data/processed/srfn_covariates_grouped.csv',

                       # also specify how site is read in
                       col_types = cols(site = col_factor())
                       )
```

Data checks

```
str(covariates)
```

```
## spc_tbl_ [1,200 x 63] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ site_number   : num [1:1200] 1 2 4 6 10 12 13 17 18 21 ...
##  $ site          : Factor w/ 60 levels "LUD_1","LUC_2",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ buff_dist     : num [1:1200] 250 250 250 250 250 250 250 250 250 250 ...
##  $ 1940          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 1950          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 1960          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 1967          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 1968          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 1969          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 1970          : num [1:1200] 0 0.342 0 0.209 0 ...
##  $ 1975          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 1976          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 1977          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 1980          : num [1:1200] 0 0 0 0 0 ...
##  $ 1983          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 1984          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 1985          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 1986          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 1987          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 1988          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 1989          : num [1:1200] 0.0285 0 0 0 0 ...
##  $ 1990          : num [1:1200] 0 0 0 0 0 ...
##  $ 1993          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 1994          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 1995          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 1996          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 1997          : num [1:1200] 0.0478 0 0 0 0 ...
##  $ 1998          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 1999          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 2000          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 2001          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 2003          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 2004          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 2005          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 2006          : num [1:1200] 0 0 0 0.179 0.424 ...
##  $ 2007          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 2008          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 2009          : num [1:1200] 0 0 0 0 0 ...
##  $ 2010          : num [1:1200] 0.355 0 0 0 0 ...
##  $ 2011          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 2012          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 2013          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 2014          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 2015          : num [1:1200] 0 0 0 0 0 ...
##  $ 2016          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 2017          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 2018          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 2019          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 2020          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 2021          : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
```

```
##  $ harvest       : num [1:1200] 0.432 0.342 0 0.388 0.424 ...
##  $ pipeline      : num [1:1200] 0 0.148 0.0148 0 0 ...
##  $ roads         : num [1:1200] 0.00 5.99e-02 7.05e-03 7.11e-06 6.75e-03 ...
##  $ seismic_lines : num [1:1200] 0.00 5.41e-05 0.00 0.00 0.00 ...
##  $ veg_edges     : num [1:1200] 0 0.09955 0.0129 0.00112 0.01425 ...
##  $ wells         : num [1:1200] 0 0 0.0183 0.0318 0.0332 ...
##  $ lc_agriculture: num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ lc_broadleaf  : num [1:1200] 0 0.18 0 0 0 ...
##  $ lc_coniferous : num [1:1200] 0.847 0 0.743 0.442 0.284 ...
##  $ lc_developed  : num [1:1200] 0 0.4514 0.0716 0.00837 0.04522 ...
##  $ lc_grassland  : num [1:1200] 0 0.3608 0.0618 0 0 ...
##  $ lc_mixed      : num [1:1200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ lc_shrub      : num [1:1200] 0.15301 0.00776 0.12401 0.54941 0.6703 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   site_number = col_double(),
##   ..   site = col_factor(levels = NULL, ordered = FALSE, include_na = FALSE),
##   ..   buff_dist = col_double(),
##   ..   `1940` = col_double(),
##   ..   `1950` = col_double(),
##   ..   `1960` = col_double(),
##   ..   `1967` = col_double(),
##   ..   `1968` = col_double(),
##   ..   `1969` = col_double(),
##   ..   `1970` = col_double(),
##   ..   `1975` = col_double(),
##   ..   `1976` = col_double(),
##   ..   `1977` = col_double(),
##   ..   `1980` = col_double(),
##   ..   `1983` = col_double(),
##   ..   `1984` = col_double(),
##   ..   `1985` = col_double(),
##   ..   `1986` = col_double(),
##   ..   `1987` = col_double(),
##   ..   `1988` = col_double(),
##   ..   `1989` = col_double(),
##   ..   `1990` = col_double(),
##   ..   `1993` = col_double(),
##   ..   `1994` = col_double(),
##   ..   `1995` = col_double(),
##   ..   `1996` = col_double(),
##   ..   `1997` = col_double(),
##   ..   `1998` = col_double(),
##   ..   `1999` = col_double(),
##   ..   `2000` = col_double(),
##   ..   `2001` = col_double(),
##   ..   `2003` = col_double(),
##   ..   `2004` = col_double(),
##   ..   `2005` = col_double(),
##   ..   `2006` = col_double(),
##   ..   `2007` = col_double(),
##   ..   `2008` = col_double(),
##   ..   `2009` = col_double(),
##   ..   `2010` = col_double(),
```

```
##   ..   `2011` = col_double(),
##   ..   `2012` = col_double(),
##   ..   `2013` = col_double(),
##   ..   `2014` = col_double(),
##   ..   `2015` = col_double(),
##   ..   `2016` = col_double(),
##   ..   `2017` = col_double(),
##   ..   `2018` = col_double(),
##   ..   `2019` = col_double(),
##   ..   `2020` = col_double(),
##   ..   `2021` = col_double(),
##   ..   harvest = col_double(),
##   ..   pipeline = col_double(),
##   ..   roads = col_double(),
##   ..   seismic_lines = col_double(),
##   ..   veg_edges = col_double(),
##   ..   wells = col_double(),
##   ..   lc_agriculture = col_double(),
##   ..   lc_broadleaf = col_double(),
##   ..   lc_coniferous = col_double(),
##   ..   lc_developed = col_double(),
##   ..   lc_grassland = col_double(),
##   ..   lc_mixed = col_double(),
##   ..   lc_shrub = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

Subset data by buffer

We do need to subset the data so we have separate data frames for each buffer width to work with in the analysis **AND** to explore correlations between variables at each buffer width, as these may very with spatial scales

Let's use a for loop to subset the data, thanks Andrew!

```
buffer_frames <- list()

for (i in unique(covariates$buff_dist)){

  print(i)

  # Subset data based on radius
  df <- covariates %>%
    filter(buff_dist == i)

  # list of dataframes
  buffer_frames <-c (buffer_frames, list(df))
}
```

```
## [1] 250
## [1] 500
## [1] 750
## [1] 1000
```

```
## [1] 1250
## [1] 1500
## [1] 1750
## [1] 2000
## [1] 2250
## [1] 2500
## [1] 2750
## [1] 3000
## [1] 3250
## [1] 3500
## [1] 3750
## [1] 4000
## [1] 4250
## [1] 4500
## [1] 4750
## [1] 5000
```

```
# name list objects so we can extract names for plotting

buffer_frames <- buffer_frames %>%

  # absurdly long way to do this but for sake of time fuck it
  purrr::set_names('250 meter buffer',
                   '500 meter buffer',
                   '750 meter buffer',
                   '1000 meter buffer',
                   '1250 meter buffer',
                   '1500 meter buffer',
                   '1750 meter buffer',
                   '2000 meter buffer',
                   '2250 meter buffer',
                   '2500 meter buffer',
                   '2750 meter buffer',
                   '3000 meter buffer',
                   '3250 meter buffer',
                   '3500 meter buffer',
                   '3750 meter buffer',
                   '4000 meter buffer',
                   '4250 meter buffer',
                   '4500 meter buffer',
                   '4750 meter buffer',
                   '5000 meter buffer')
```

Now we have a list with data frames for each buffer width which we can work with later.

Add response metrics

Now that we have the covariate data formatted we need to the response metric data frames that we will use for each species and tidy up the resulting data

We will want separate data frames for each type of response metric - we will use the proportional detection data for all the species we have enough detections for, otherwise we will use either the total detections or presence absence data depending how the models fit.

```
# proportional detections
prop_det_data <- buffer_frames %>%

  # use purrr to join data to all individual buffer frames data sets
  purrr::map(
    ~.x %>%

      # use left join so only sites with covariate data are kept
      left_join(response_metrics_subset$prop_detections,
                by = 'site'))


# total detections
total_det_data <- buffer_frames %>%

  # use purrr to join data to all individual buffer frames data sets
  purrr::map(
    ~.x %>%

      # use left join so only sites with covariate data are kept
      left_join(response_metrics_subset$total_detections,
                by = 'site'))

# presence absence
pres_absen_dat <-  buffer_frames %>%

  # use purrr to join data to all individual buffer frames data sets
  purrr::map(
    ~.x %>%

      # use left join so only sites with covariate data are kept
      left_join(response_metrics_subset$presence_absence,
                by = 'site'))
```

I'm going to view each of these through the RStudio environemnt to look at them in depth and will print a subset of each ehre

```
head(prop_det_data$`250 meter buffer`)
```

```
## # A tibble: 6 x 79
##   site_number site    buff_dist `1940` `1950` `1960` `1967` `1968` `1969` `1970`
##         <dbl> <fct>       <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
## 1           1 LUD_1         250      0      0      0      0      0      0  0
## 2           2 LUC_2         250      0      0      0      0      0      0  0.342
## 3           4 LUC_4         250      0      0      0      0      0      0  0
## 4           6 LUS_6         250      0      0      0      0      0      0  0.209
## 5          10 LUS_10        250      0      0      0      0      0      0  0
## 6          12 LUBF_12       250      0      0      0      0      0      0  0
## # i 69 more variables: `1975` <dbl>, `1976` <dbl>, `1977` <dbl>, `1980` <dbl>,
## #   `1983` <dbl>, `1984` <dbl>, `1985` <dbl>, `1986` <dbl>, `1987` <dbl>,
## #   `1988` <dbl>, `1989` <dbl>, `1990` <dbl>, `1993` <dbl>, `1994` <dbl>,
## #   `1995` <dbl>, `1996` <dbl>, `1997` <dbl>, `1998` <dbl>, `1999` <dbl>,
```

```
## #   `2000` <dbl>, `2001` <dbl>, `2003` <dbl>, `2004` <dbl>, `2005` <dbl>,
## #   `2006` <dbl>, `2007` <dbl>, `2008` <dbl>, `2009` <dbl>, `2010` <dbl>,
## #   `2011` <dbl>, `2012` <dbl>, `2013` <dbl>, `2014` <dbl>, `2015` <dbl>, ...
```

Finish with data formatting

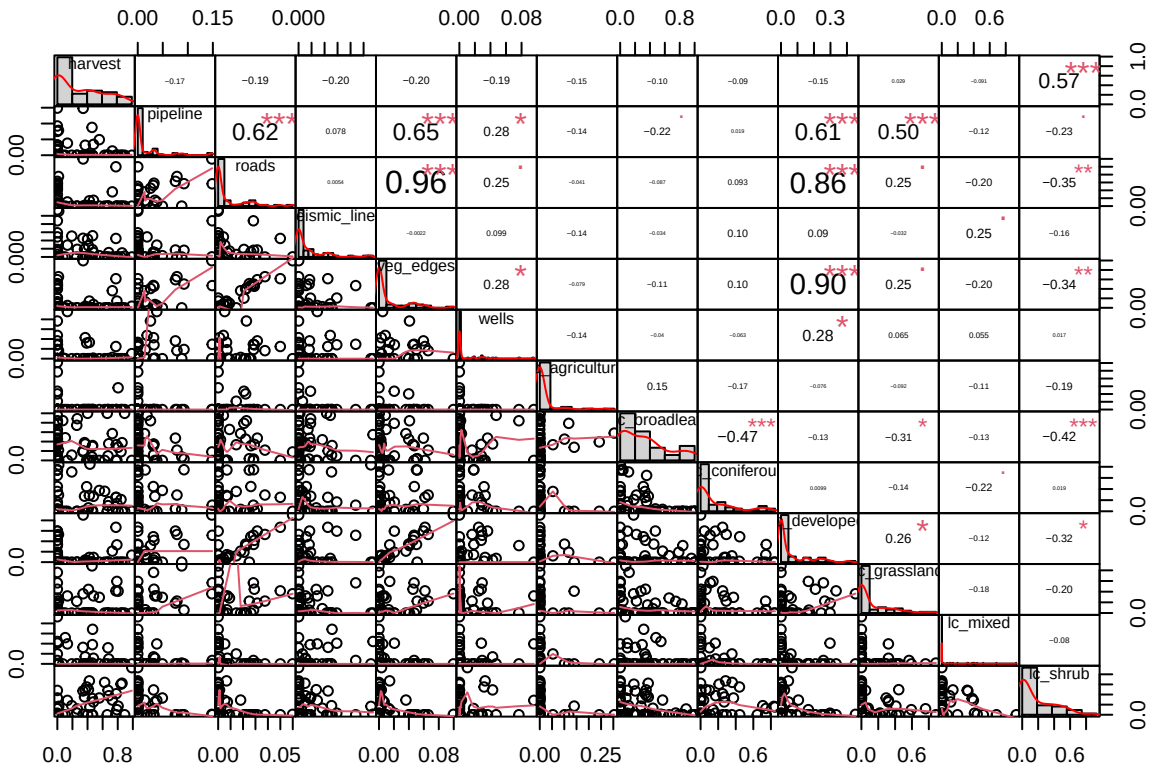
Let's remove the objects we no longer need from the environment to keep our work space clean

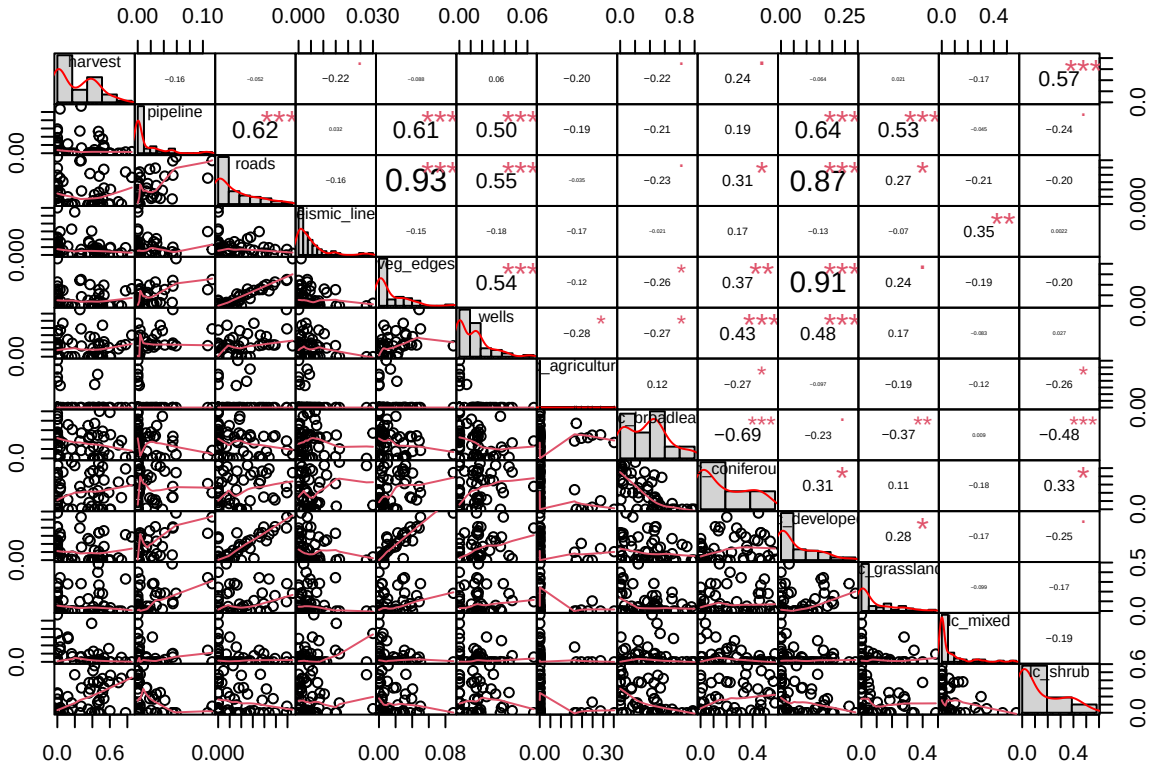
```
rm(buffer_frames,
   covariates,
   df,
   response_metrics_subset)
```

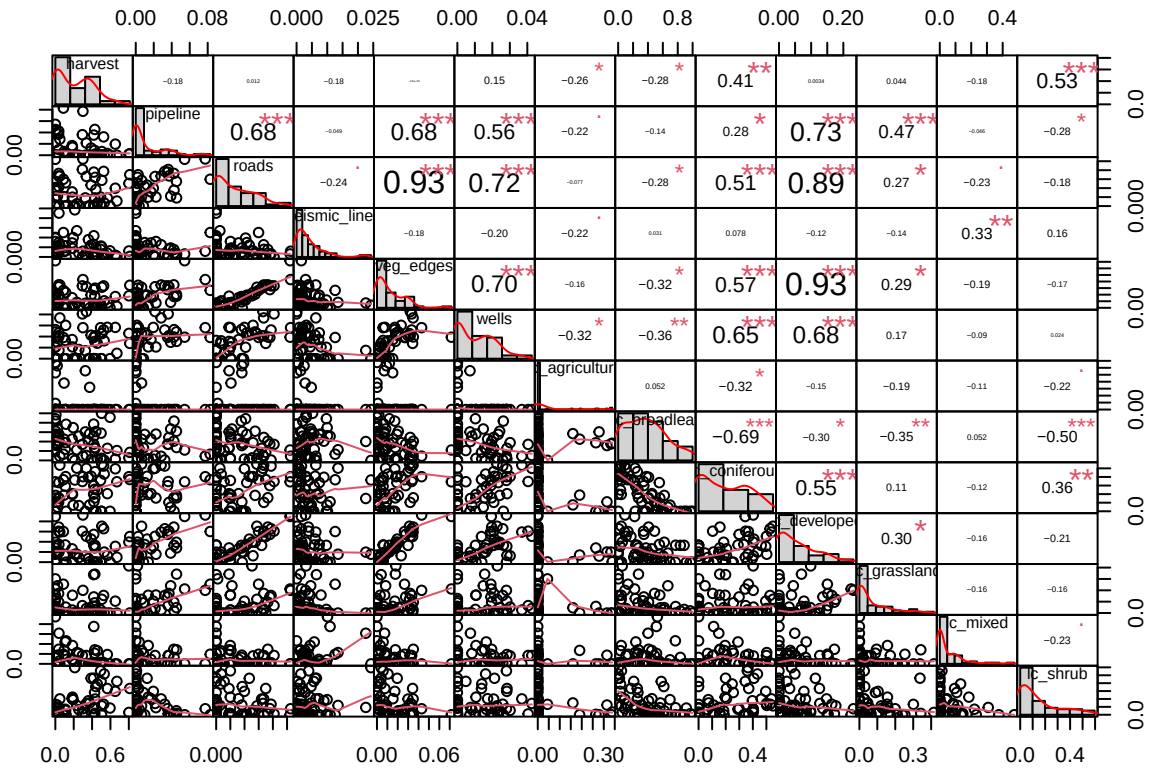
Correlation plots

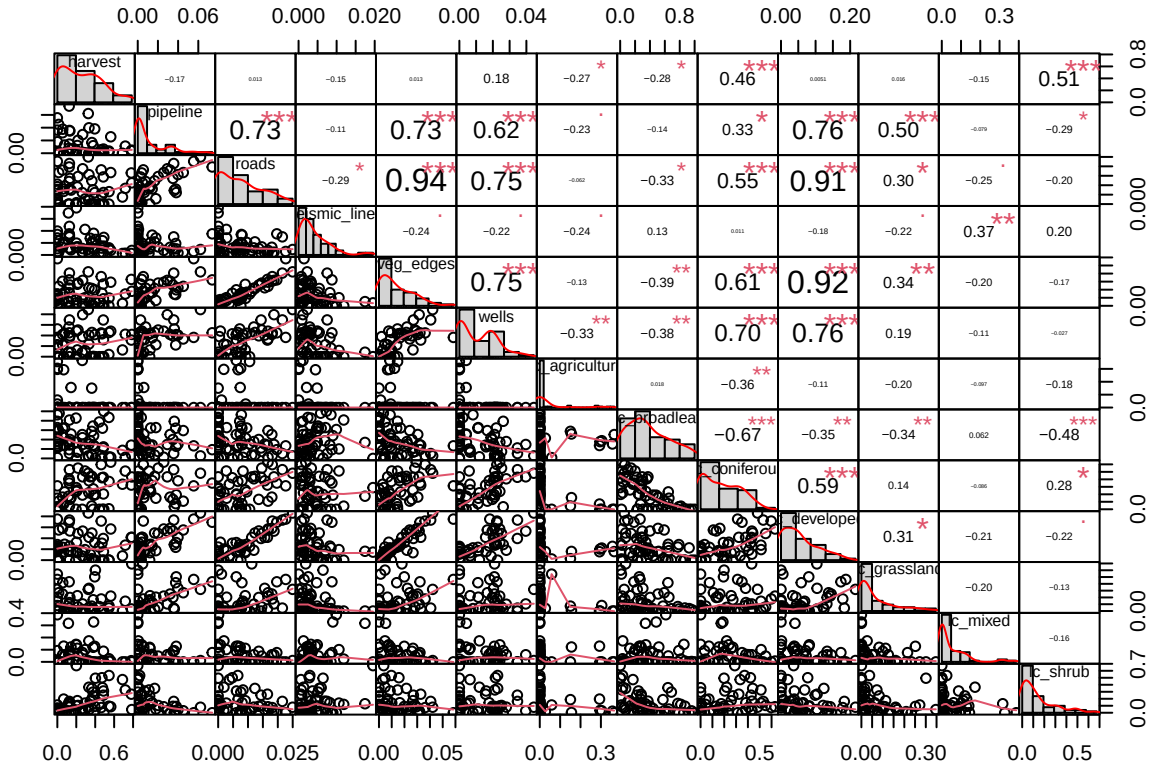
Before we can develop any models we need to look at your covariates and see if there are any major violations of the model assumptions of independence, e.g. variables that are highly correlated - in which case we need to choose which variable in each pair to include in a model

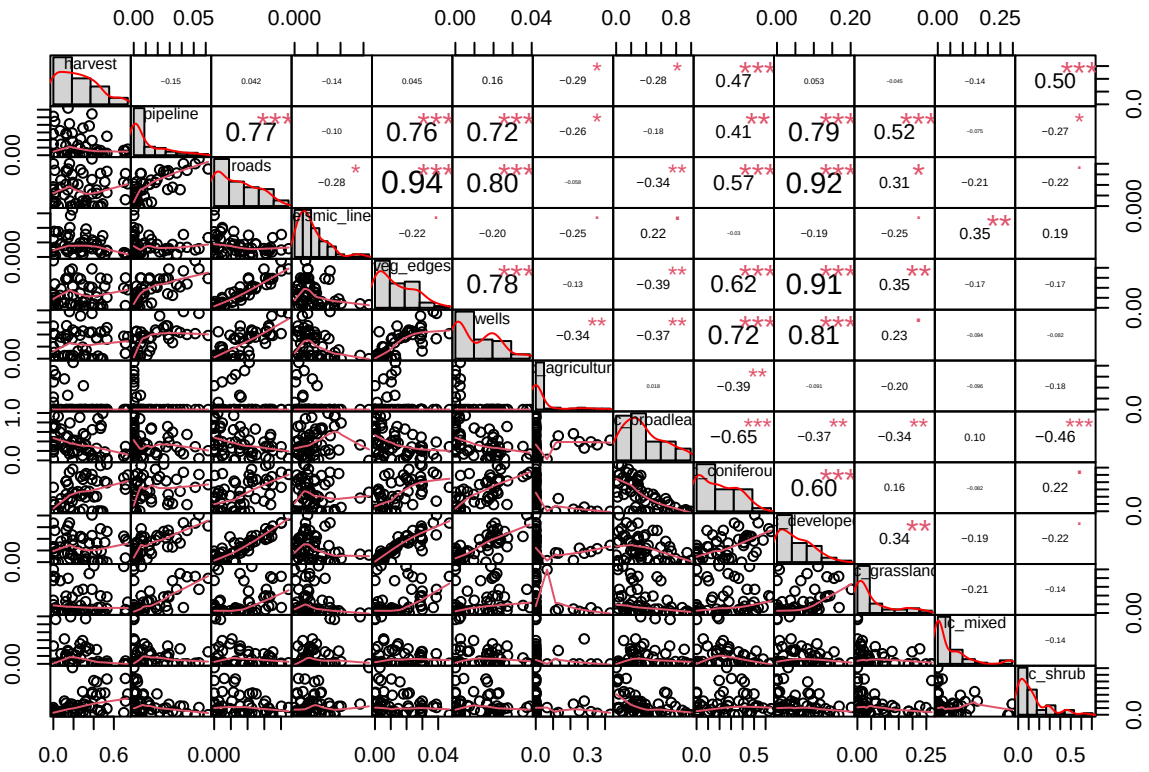
```
prop_det_data %>%

  purrr::map(
    ~.x %>%

      # select only columns with covariates not other info to simplify the plot a bit
      select(harvest:lc_shrub) %>%

      # use chart.correlation to produce plots for each buffer size
      chart.Correlation(.,
                        histogram = TRUE,
                        method = "pearson")
  )
```

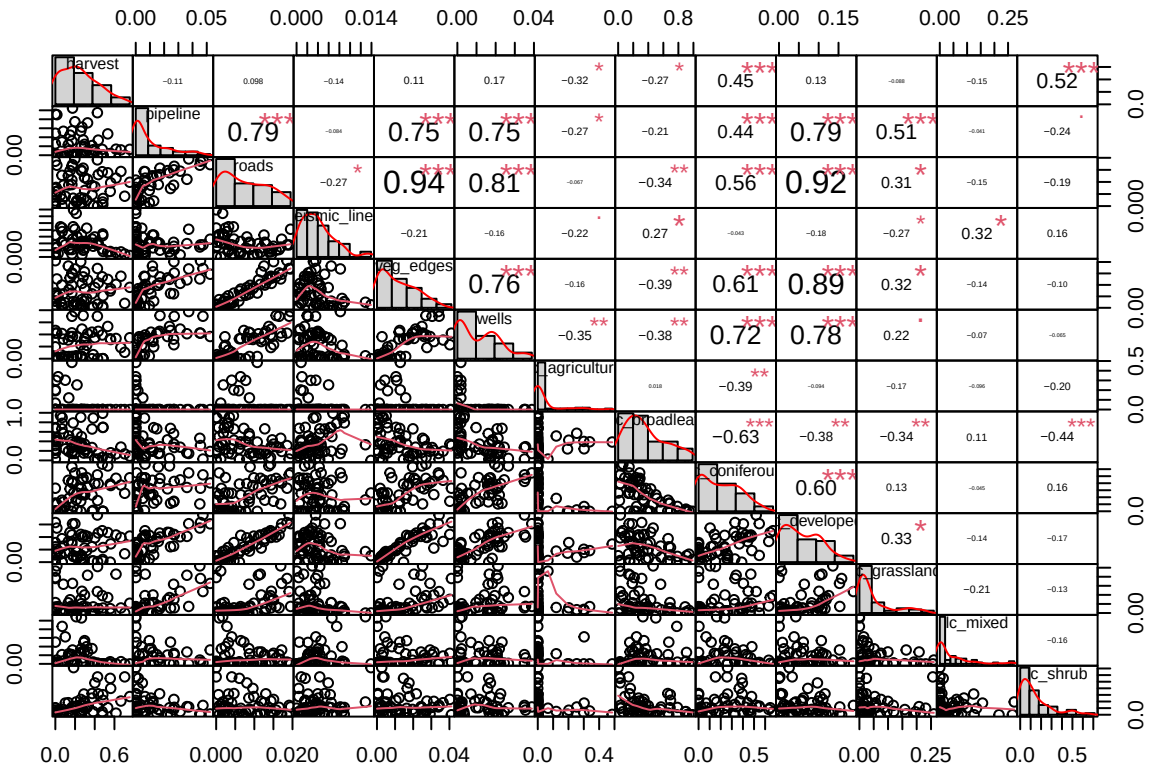

harvest
pipeline
roads
seismic_line
veg_edges
wells
agricultur
broadlea
coniferou
develope
grassland
lc_mixed
lc_shrub

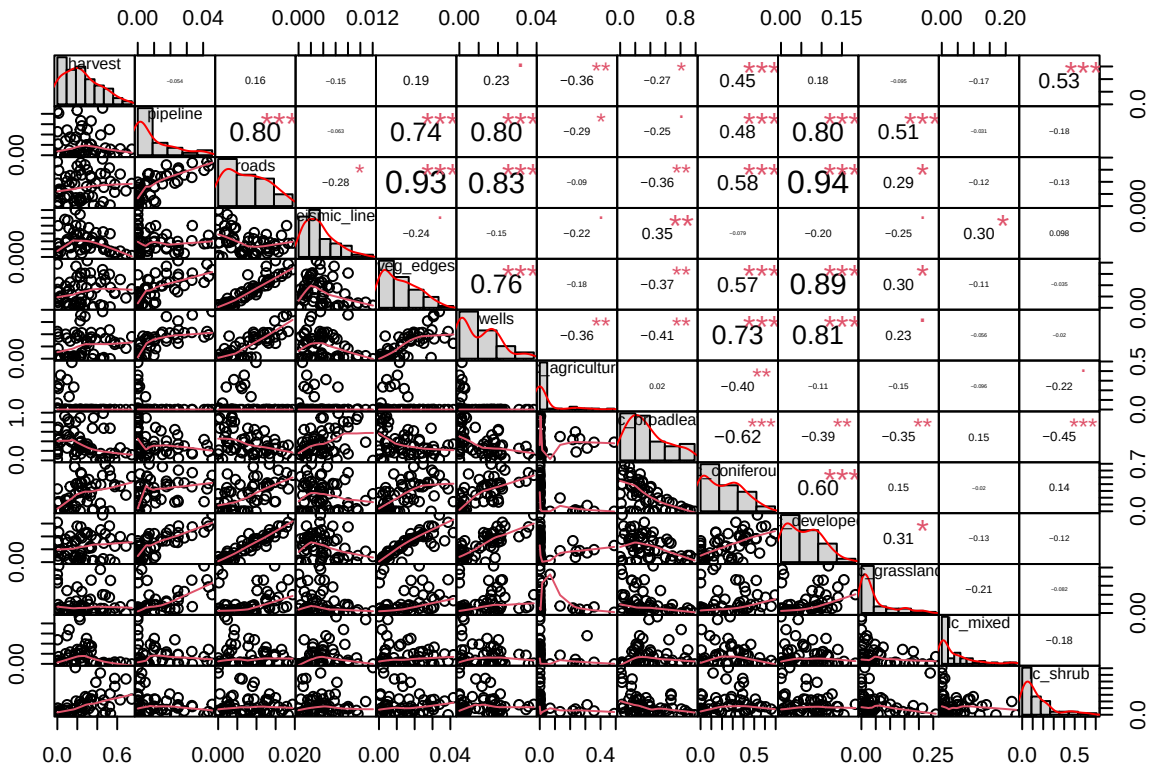

harvest
pipeline
roads
seismic_line
veg_edges
wells
c_agricultur
c_broadlea
c_coniferou
develope
c_grassland
c_mixed
c_shrub

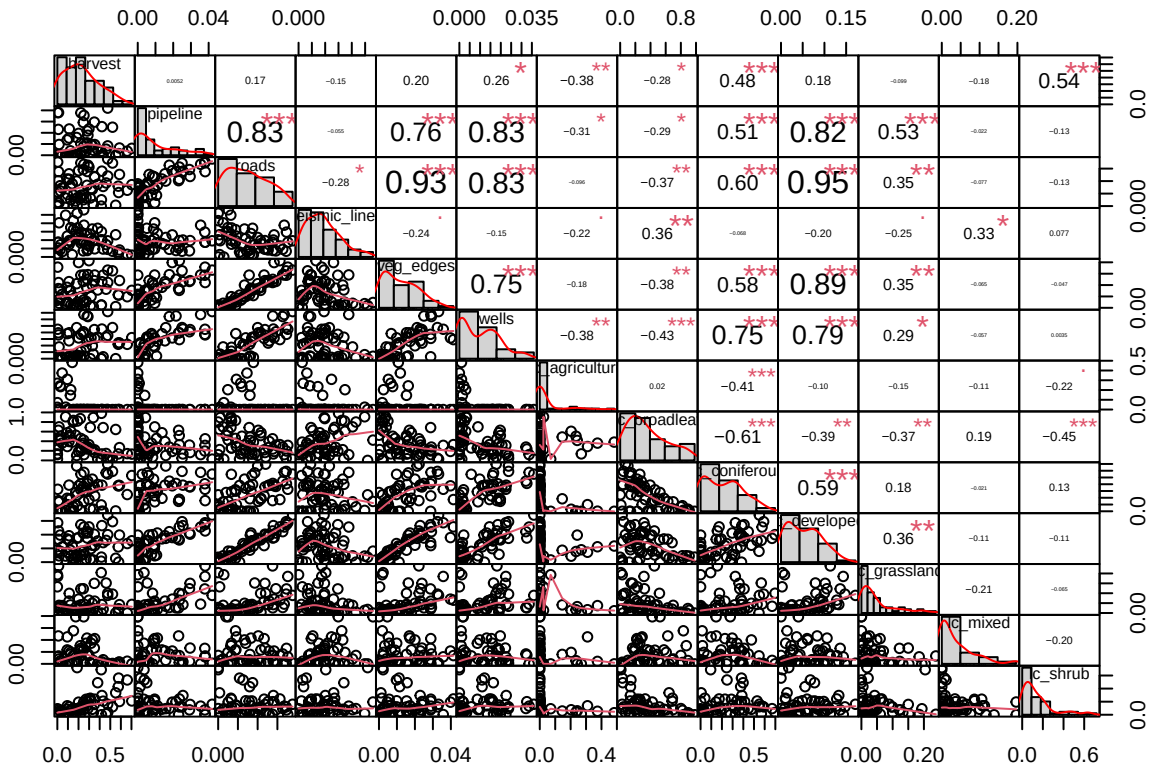

harvest
pipeline
roads
seismic_line
veg_edges
wells
c_agricultur
c_broadlea
c_coniferou
c_develope
c_grassland
c_mixed
c_shrub

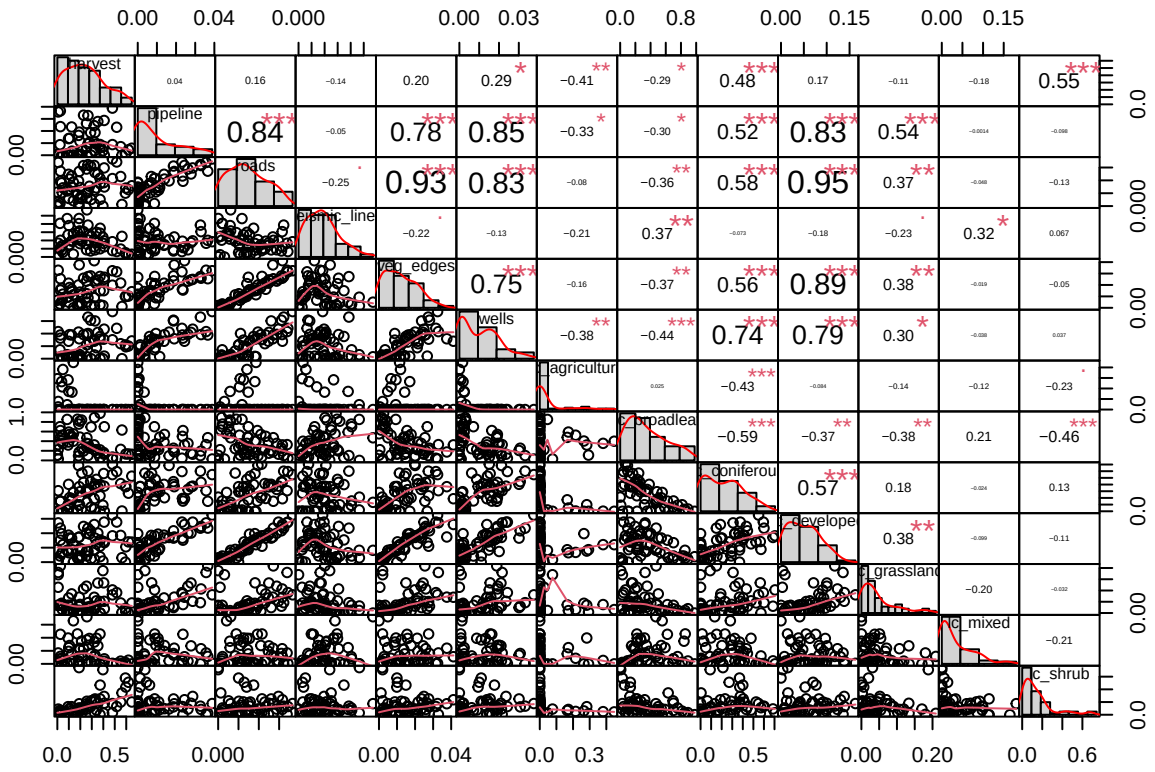

harvest
pipeline
roads
seismic_line
veg_edges
wells
agricultur
broadlea
coniferou
develope
grassland
c_mixed
c_shrub

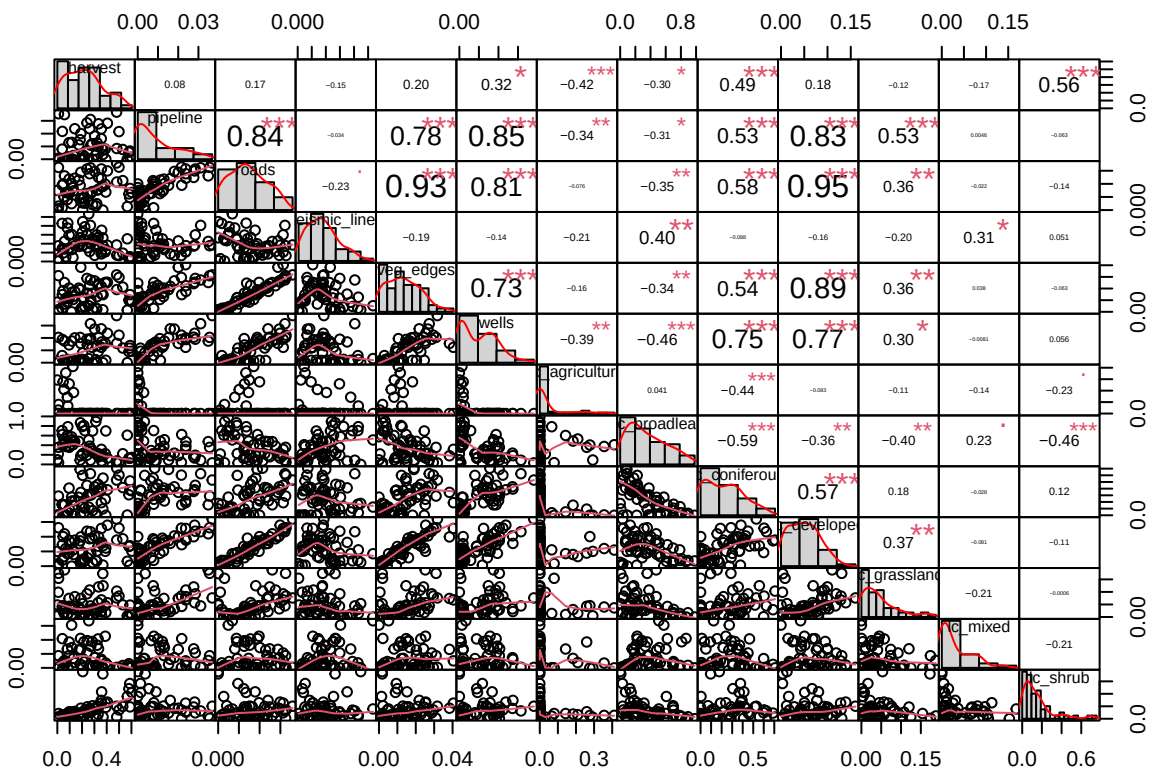

harvest
pipeline
roads
seismic_line
veg_edges
wells
agricultur
broadlea
coniferou
develope
grassland
c_mixed
c_shrub

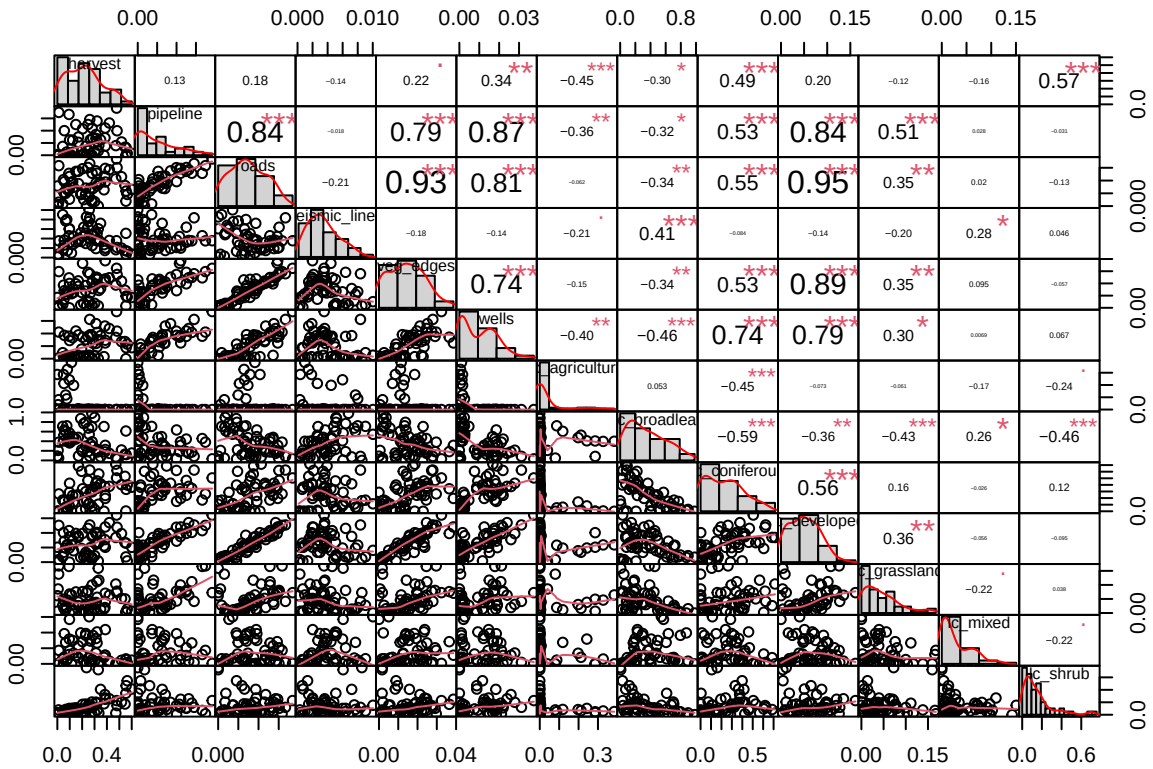

harvest
pipeline
roads
seismic_line
veg_edges
wells
agricultur
cropadlea
coniferou
develope
grassland
lc_mixed
lc_shrub

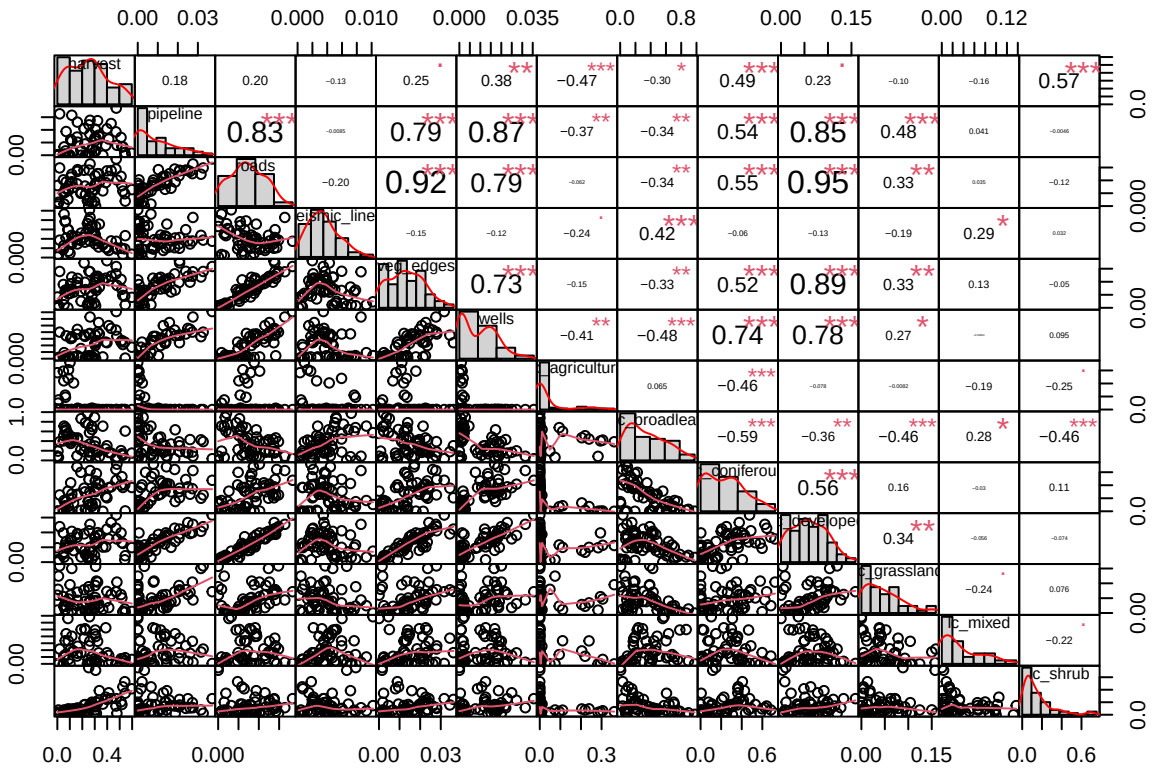

harvest
pipeline
roads
seismic_line
veg_edges
wells
agricultur
cropadlea
coniferou
develope
grassland
c_mixed
c_shrub

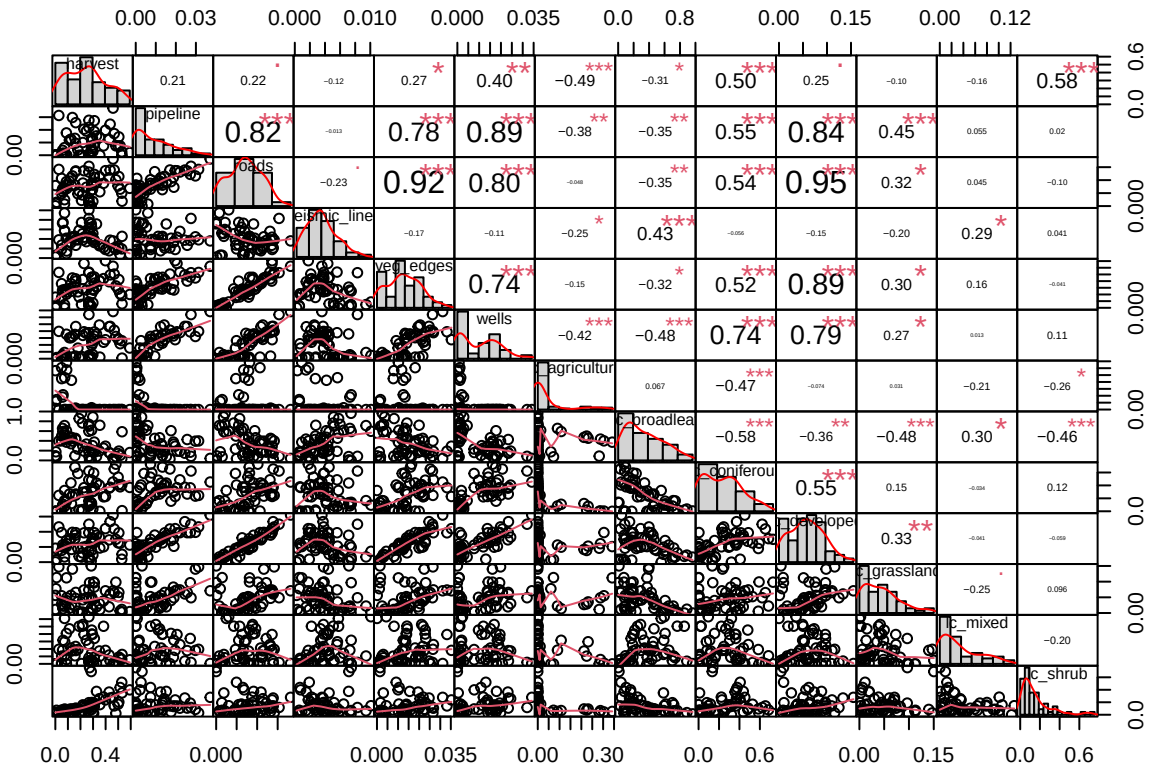

harvest
pipeline
roads
seismic_line
veg_edges
wells
agricultur
cropdlea
coniferou
develope
c_grassland
c_mixed
c_shrub

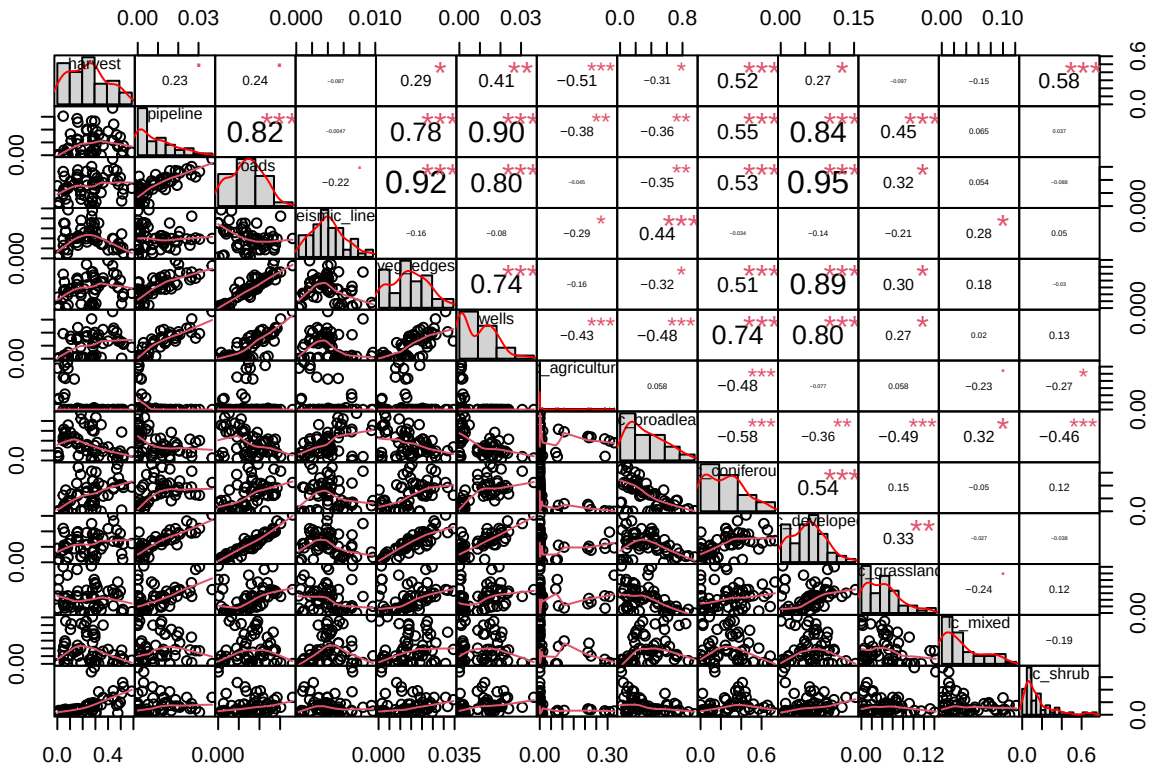

harvest
pipeline
roads
trans_line
veg_edges
wells
agricultur
cropdlea
coniferou
develope
c_grassland
c_mixed
c_shrub

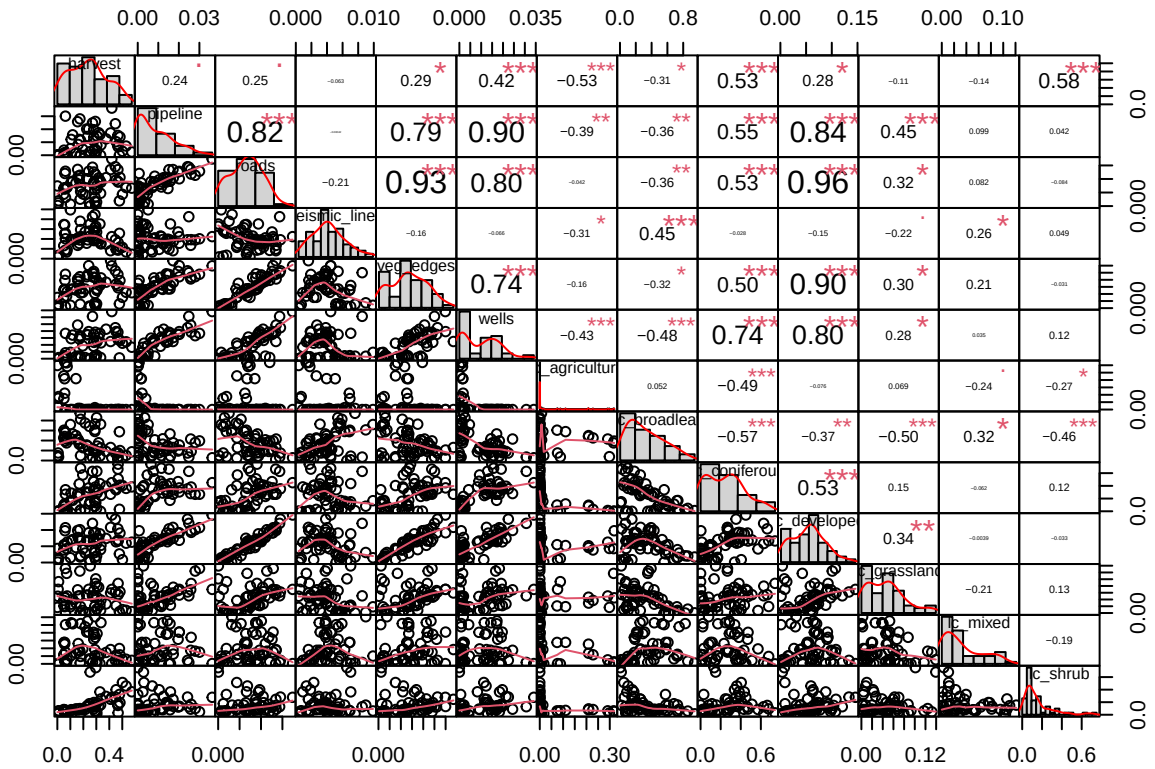

harvest
pipeline
roads
seismic_line
pre_edges
wells
agricultur
broadlea
coniferou
develope
grassland
mixed
shrub

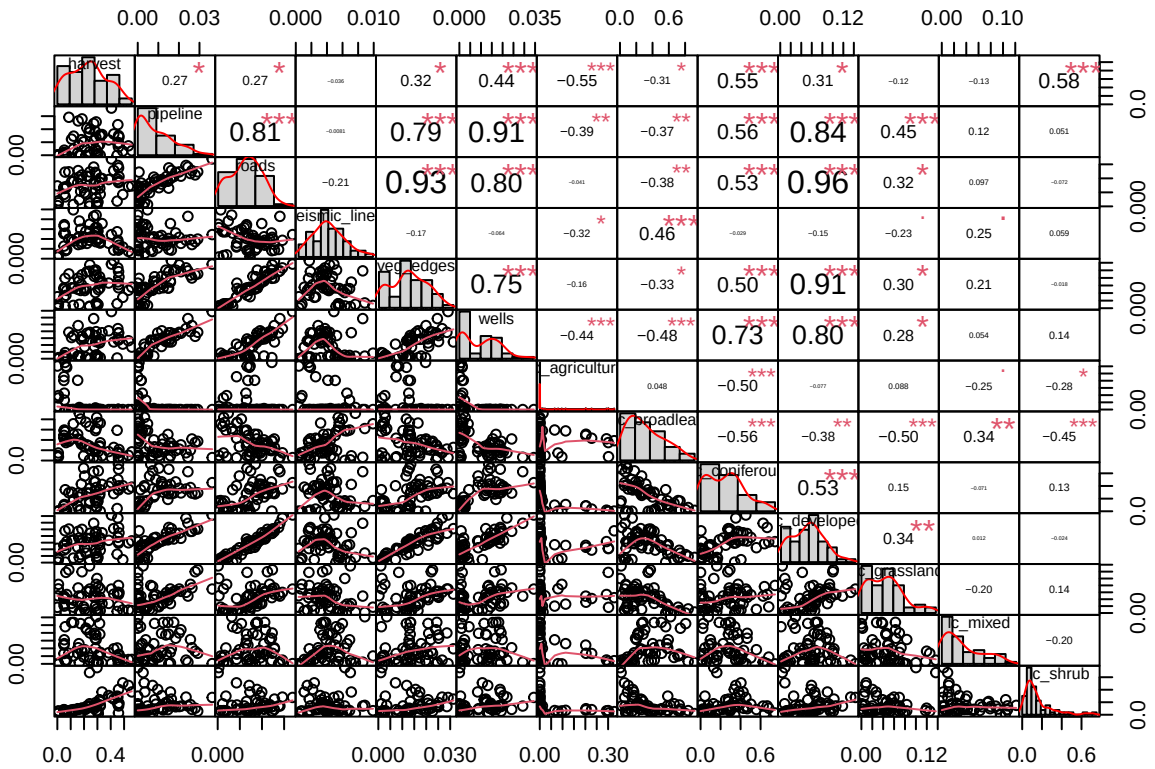

harvest
pipeline
roads
seismic_line
reg_edges
wells
agricultur
broadlea
coniferou
develope
grassland
lc_mixed
lc_shrub

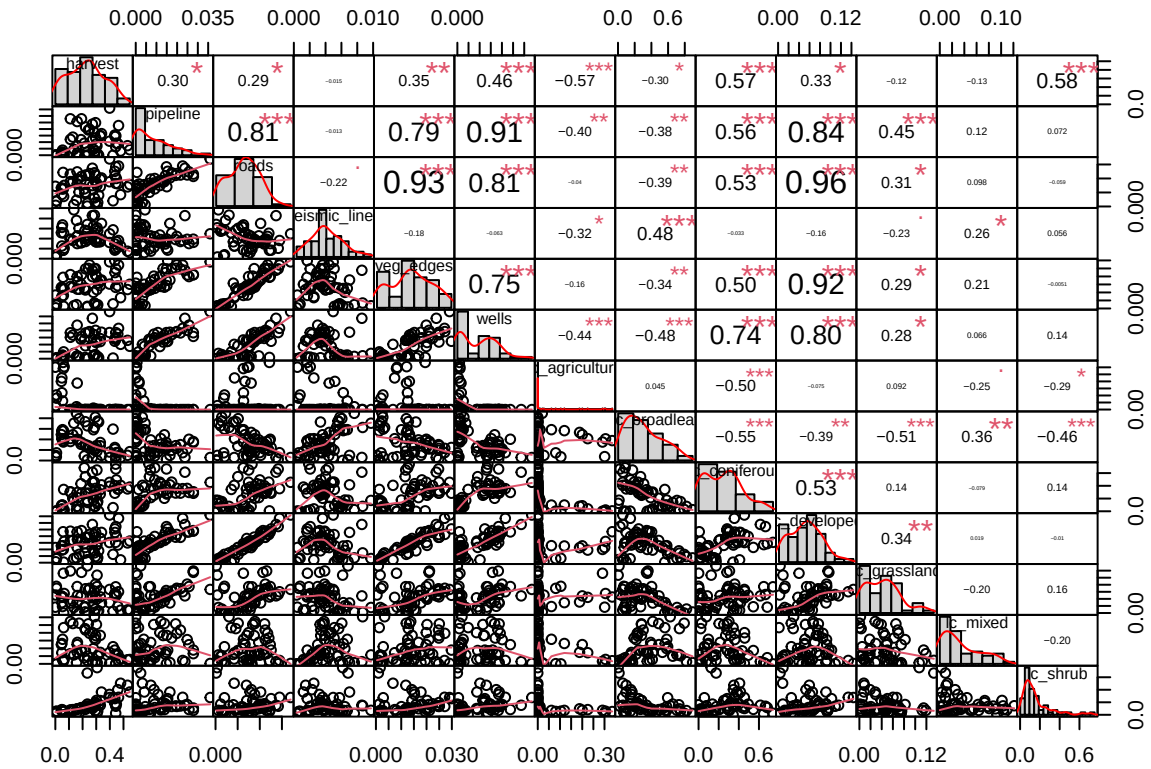

harvest
pipeline
roads
seismic_line
well_edges
wells
agricultur
broadlea
coniferou
develope
grassland
lc_mixed
lc_shrub

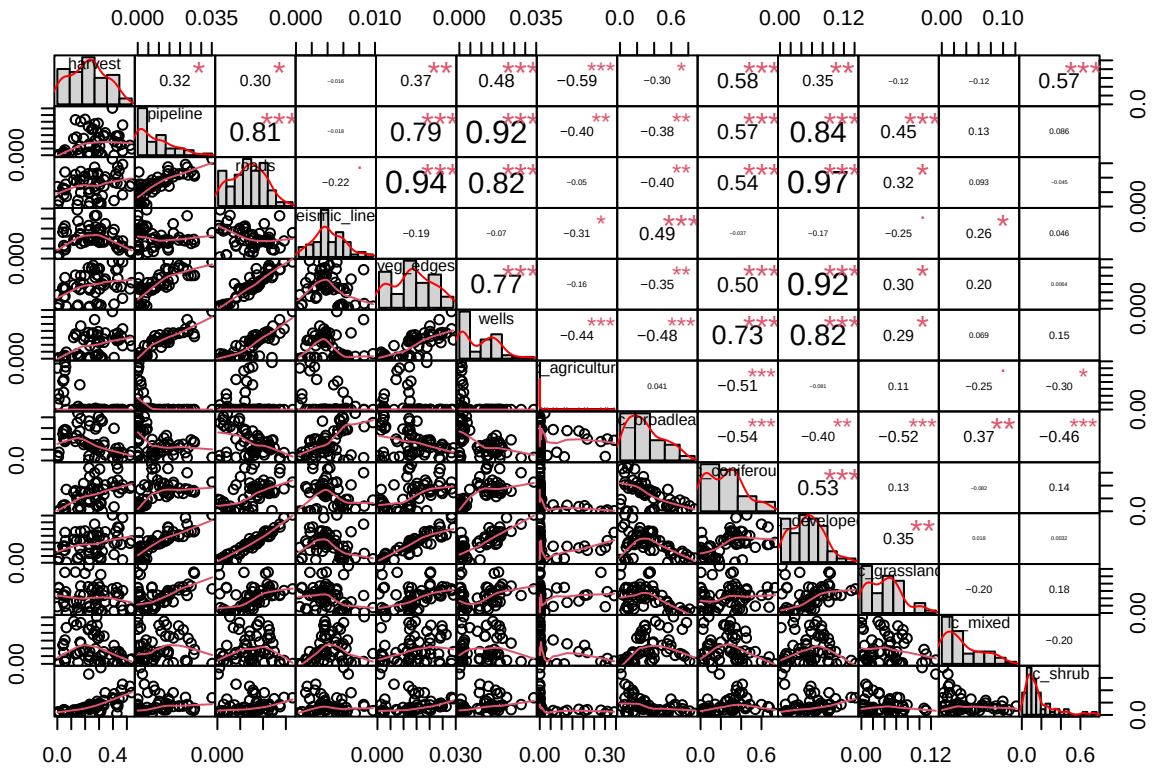

harvest
pipeline
roads
seismic_line
veg_edges
wells
agricultur
broadlea
coniferou
develope
grassland
lc_mixed
lc_shrub

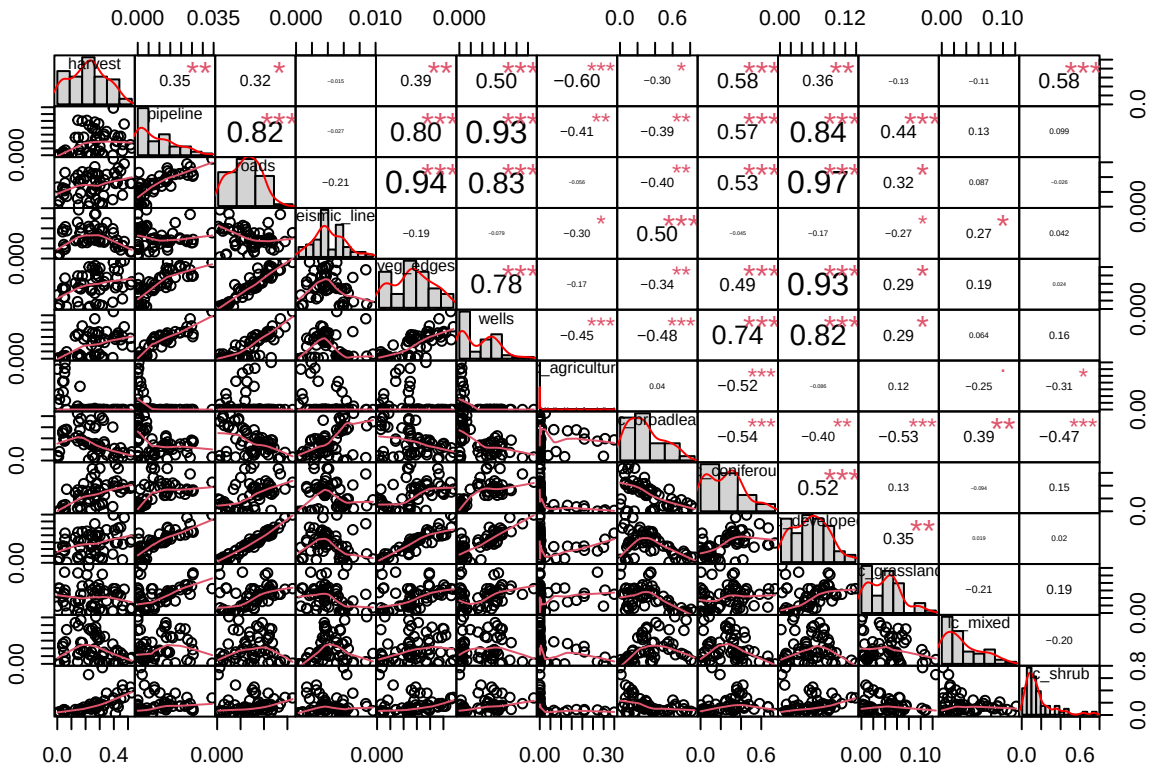

harvest
pipeline
roads
seismic_line
veg_edges
wells
c_agricultur
broadlea
coniferou
develope
grassland
lc_mixed
lc_shrub

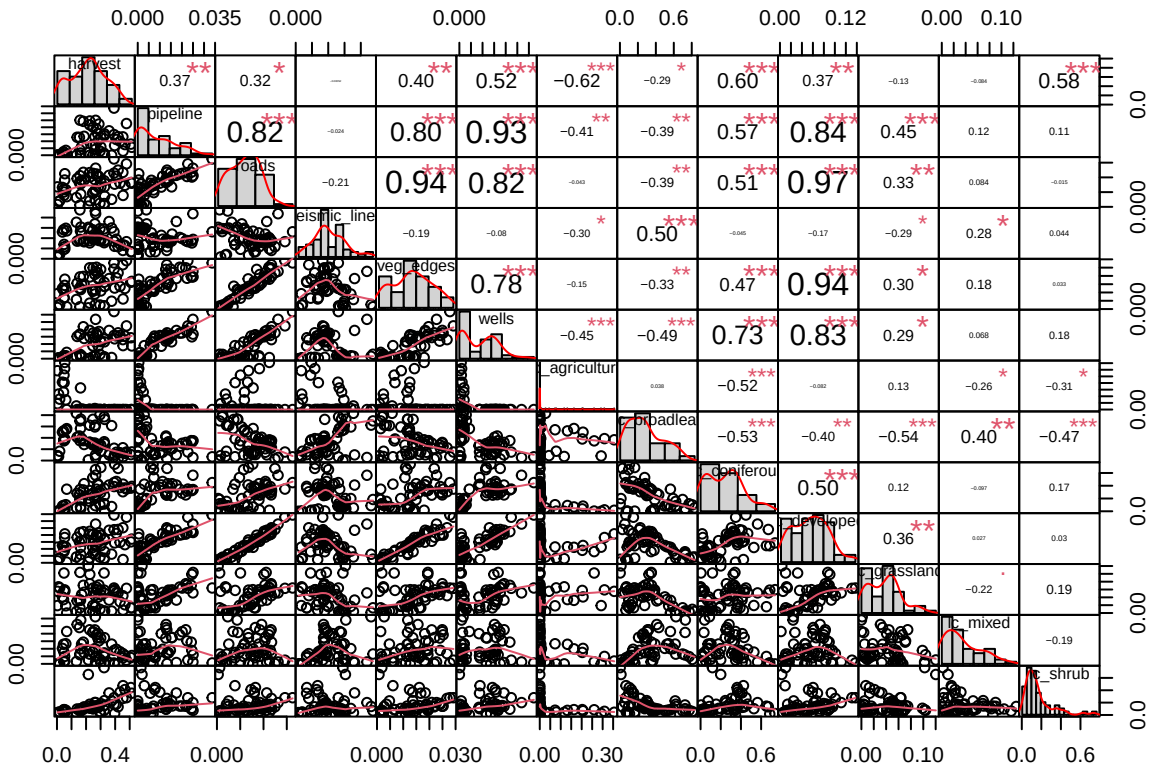

harvest
pipeline
roads
seismic_line
veg_edges
wells
c_agricultur
broadlea
coniferou
develope
grassland
lc_mixed
lc_shrub


harvest
pipeline
roads
seismic_line
veg_edges
wells
c_agricultur
broadlea
coniferou
develope
grassland
lc_mixed
lc_shrub


harvest
pipeline
roads
seismic_line
veg_edges
wells
c_agricultur
c_broadlea
c_coniferou
c_develope
c_grassland
c_mixed
c_shrub


harvest
pipeline
roads
seismic_line
veg_edges
wells
c_agricultur
cropdadlea
conferou
developed
c_grassland
c_mixed
c_shrub


harvest
pipeline
roads
seismic_line
veg_edges
wells
c_agricultur
cropdadlea
coniferou
developed
grassland
c_mixed
c_shrub




```
## $`250 meter buffer`
## NULL
##
## $`500 meter buffer`
## NULL
##
## $`750 meter buffer`
## NULL
##
## $`1000 meter buffer`
## NULL
##
## $`1250 meter buffer`
## NULL
##
## $`1500 meter buffer`
## NULL
##
## $`1750 meter buffer`
## NULL
##
## $`2000 meter buffer`
## NULL
##
## $`2250 meter buffer`
## NULL
```

```
##
## $`2500 meter buffer`
## NULL
##
## $`2750 meter buffer`
## NULL
##
## $`3000 meter buffer`
## NULL
##
## $`3250 meter buffer`
## NULL
##
## $`3500 meter buffer`
## NULL
##
## $`3750 meter buffer`
## NULL
##
## $`4000 meter buffer`
## NULL
##
## $`4250 meter buffer`
## NULL
##
## $`4500 meter buffer`
## NULL
##
## $`4750 meter buffer`
## NULL
##
## $`5000 meter buffer`
## NULL
```

Summary of correlation plots per buffer

Here I summarize any pairs of highly correlated variables

Overall

- several things correlated with veg edges which we expected so will drop veg edges
- several HFI features and lc_developed, also expected will use the features when possible as they are more specific/informative

- roads correlated with wells and pipelines - also expected you need roads to access/build these features
- lc_broadleaf and lc_coniferous also expected as they are the main landcover types and tend to be very inversely related, for each species will choose based on ecology

Buffer specific deviations from overall

750 m buffer * roads and wells (continues as buffer size increases)

1000m buffer
* wells and lc_coniferous are positively correlated starting around 750m buffer - kind of a weird one maybe?

1250m buffer
* pipeline and wells (starts here 0.72 and continues with increasing buffer size)

So based on this the global model for buffer size selection should generally include the following set of variables harvest + roads + seismic_lines + lc_agriculture +
lc_broadleaf + lc_grassland + lc_mixed + lc_shrub,

Black bear

Buffer selection

First we need to check which buffer size is most informative for each species, we do this by running the same global model repeatedly but with different buffer sizes, for each species.

Let's start with bears and use purrr to create a global model for every buffer distance

Recall `purrr::map()` is magical for iterations and will apply all the functions within the `map()` function to each item of the list supplied before the the `map()` function.

```
# create models for black bears at each buffer size
black_bear_buff <-  prop_det_data %>%

  # use purrr map to run the same functions for all buffer sizes ((all objects in list))
  purrr::map(
    ~.x %>%

      # glm function let's us run the proportional binomial model using cbind to combine the present an
      glm(cbind(black_bear, absent_black_bear) ~

            # covariates not correlated
            harvest +
            roads +
            seismic_lines +
            lc_agriculture +
            lc_broadleaf  +
            lc_grassland  +
            lc_mixed  +
            lc_shrub,

          data = .,
          family = 'binomial'))

model.sel(black_bear_buff)
```

```
## Model selection table
##                     (Int)     hrv   lc_agr  lc_brd  lc_grs   lc_mxd  lc_shr
## 250 meter buffer  -1.851 -0.6106  0.31760  0.6844  1.6310  0.01112  1.1480
## 4750 meter buffer -3.746  3.2870  4.57600 -1.0980  6.0830 10.66000 -2.1810
## 4500 meter buffer -3.612  2.9060  4.33100 -1.1310  4.4290  9.99300 -1.9970
## 5000 meter buffer -3.802  3.5170  4.76700 -1.0120  7.3940 10.86000 -2.2600
## 4250 meter buffer -3.457  2.4810  3.85600 -1.0410  3.5710  8.74200 -1.8460
```

```
## 3500 meter buffer -2.794  2.3920  3.28200 -1.3820  3.0780  7.12300 -2.2690
## 3750 meter buffer -3.042  2.2280  3.39400 -1.1860  2.2210  7.46100 -1.9270
## 4000 meter buffer -3.232  2.1280  3.33300 -0.9778  2.5460  7.84600 -1.7280
## 3250 meter buffer -2.595  2.2620  2.76800 -1.2780  2.5710  5.75200 -2.2120
## 3000 meter buffer -2.450  2.0540  2.33800 -1.1040  1.9580  4.47500 -1.9840
## 2750 meter buffer -2.371  1.8840  1.99200 -0.9989  1.5240  3.50400 -1.8270
## 2250 meter buffer -2.122  1.7470  1.50600 -0.8485  0.2720  1.53300 -1.8490
## 2500 meter buffer -2.222  1.6590  1.61400 -0.8723  0.2691  2.44500 -1.7020
## 2000 meter buffer -1.889  1.6260  1.13400 -0.8362 -0.6038  0.48240 -1.8450
## 1750 meter buffer -1.517  1.1540  0.53790 -0.7996 -1.8710 -0.24610 -1.5220
## 500 meter buffer  -2.279  0.1289  1.20600  0.8653  1.5210 -0.23250  1.5440
## 1250 meter buffer -1.008  0.7548 -0.34040 -0.7858 -3.2550 -1.45400 -1.5130
## 1000 meter buffer -1.110  1.0100  0.02604 -0.8189 -2.1350 -1.54700 -1.6630
## 1500 meter buffer -1.266  0.7353 -0.05452 -0.5764 -2.7750 -1.08900 -1.2100
## 750 meter buffer  -1.962  0.7417  0.93040  0.1138 -0.3388  0.04300  0.1525
##                       rds ssm_lns df   logLik  AICc delta weight
## 250 meter buffer  -17.820 -77.230  9 -119.272 260.2  0.00  0.332
## 4750 meter buffer  11.690 243.500  9 -119.970 261.6  1.40  0.165
## 4500 meter buffer  16.430 248.000  9 -120.043 261.8  1.54  0.153
## 5000 meter buffer   6.952 233.000  9 -120.409 262.5  2.28  0.106
## 4250 meter buffer  21.280 240.600  9 -120.499 262.7  2.45  0.097
## 3500 meter buffer -16.830 235.900  9 -121.314 264.3  4.08  0.043
## 3750 meter buffer   6.977 233.500  9 -121.371 264.4  4.20  0.041
## 4000 meter buffer  21.920 223.000  9 -121.492 264.7  4.44  0.036
## 3250 meter buffer -18.750 214.500  9 -122.235 266.1  5.93  0.017
## 3000 meter buffer -17.480 189.900  9 -123.530 268.7  8.52  0.005
## 2750 meter buffer -13.780 178.500  9 -124.189 270.1  9.84  0.002
## 2250 meter buffer -14.550 152.200  9 -125.196 272.1 11.85  0.001
## 2500 meter buffer  -6.544 154.000  9 -125.314 272.3 12.08  0.001
## 2000 meter buffer -20.540 134.400  9 -126.005 273.7 13.47  0.000
## 1750 meter buffer -25.460  94.930  9 -127.337 276.3 16.13  0.000
## 500 meter buffer   -1.354 -29.170  9 -127.578 276.8 16.61  0.000
## 1250 meter buffer -26.030  31.150  9 -127.889 277.5 17.23  0.000
## 1000 meter buffer -26.190  34.100  9 -127.951 277.6 17.36  0.000
## 1500 meter buffer -21.250  43.740  9 -128.580 278.8 18.62  0.000
## 750 meter buffer    2.991   5.566  9 -129.879 281.4 21.22  0.000
## Models ranked by AICc(x)
```

```
# just testing here how different the results are if we use glmmTMB function instead of glm

black_bear_buff_TMB <-  prop_det_data %>%

  # use purrr map to run the same functions for all buffer sizes ((all objects in list))
  purrr::map(
    ~.x %>%

      # glm function let's us run the proportional binomial model using cbind to combine the present an
      glmmTMB(cbind(black_bear, absent_black_bear) ~

            # covariates not correlated
            harvest +
            roads +
            seismic_lines +
            lc_agriculture +
```

```
            lc_broadleaf  +
            lc_grassland  +
            lc_mixed  +
            lc_shrub,

          data = .,
          family = 'binomial'))

model.sel(black_bear_buff_TMB)
```

```
## Model selection table
##                    cnd((Int)) dsp((Int)) cnd(hrv) cnd(lc_agr) cnd(lc_brd)
## 250 meter buffer       -1.851          +  -0.6106     0.31760      0.6844
## 4750 meter buffer      -3.746          +   3.2870     4.57600     -1.0980
## 4500 meter buffer      -3.612          +   2.9060     4.33100     -1.1310
## 5000 meter buffer      -3.802          +   3.5170     4.76700     -1.0120
## 4250 meter buffer      -3.457          +   2.4810     3.85600     -1.0410
## 3500 meter buffer      -2.794          +   2.3920     3.28200     -1.3820
## 3750 meter buffer      -3.042          +   2.2280     3.39400     -1.1860
## 4000 meter buffer      -3.232          +   2.1280     3.33300     -0.9778
## 3250 meter buffer      -2.595          +   2.2620     2.76800     -1.2780
## 3000 meter buffer      -2.450          +   2.0540     2.33800     -1.1040
## 2750 meter buffer      -2.371          +   1.8840     1.99200     -0.9989
## 2250 meter buffer      -2.122          +   1.7470     1.50600     -0.8485
## 2500 meter buffer      -2.222          +   1.6590     1.61400     -0.8723
## 2000 meter buffer      -1.889          +   1.6260     1.13400     -0.8362
## 1750 meter buffer      -1.517          +   1.1540     0.53790     -0.7996
## 500 meter buffer       -2.279          +   0.1289     1.20600      0.8653
## 1250 meter buffer      -1.008          +   0.7548    -0.34040     -0.7858
## 1000 meter buffer      -1.110          +   1.0100     0.02603     -0.8189
## 1500 meter buffer      -1.266          +   0.7353    -0.05452     -0.5764
## 750 meter buffer       -1.962          +   0.7417     0.93040      0.1138
##                    cnd(lc_grs) cnd(lc_mxd) cnd(lc_shr) cnd(rds) cnd(ssm_lns) df
## 250 meter buffer        1.6310     0.01112      1.1480  -17.820      -77.230  9
## 4750 meter buffer       6.0830    10.66000     -2.1810   11.680      243.500  9
## 4500 meter buffer       4.4290     9.99300     -1.9970   16.430      248.000  9
## 5000 meter buffer       7.3940    10.86000     -2.2600    6.952      233.000  9
## 4250 meter buffer       3.5710     8.74200     -1.8460   21.280      240.600  9
## 3500 meter buffer       3.0780     7.12300     -2.2690  -16.830      235.900  9
## 3750 meter buffer       2.2210     7.46100     -1.9270    6.977      233.500  9
## 4000 meter buffer       2.5460     7.84700     -1.7280   21.920      223.000  9
## 3250 meter buffer       2.5710     5.75200     -2.2120  -18.750      214.500  9
## 3000 meter buffer       1.9580     4.47500     -1.9840  -17.480      189.900  9
## 2750 meter buffer       1.5240     3.50400     -1.8270  -13.780      178.500  9
## 2250 meter buffer       0.2720     1.53300     -1.8490  -14.550      152.200  9
## 2500 meter buffer       0.2691     2.44500     -1.7020   -6.544      154.000  9
## 2000 meter buffer      -0.6038     0.48240     -1.8450  -20.540      134.400  9
## 1750 meter buffer      -1.8710    -0.24610     -1.5220  -25.460       94.930  9
## 500 meter buffer        1.5210    -0.23250      1.5440   -1.354      -29.170  9
## 1250 meter buffer      -3.2550    -1.45400     -1.5130  -26.030       31.150  9
## 1000 meter buffer      -2.1350    -1.54700     -1.6630  -26.190       34.100  9
## 1500 meter buffer      -2.7750    -1.08900     -1.2100  -21.250       43.740  9
## 750 meter buffer       -0.3388     0.04300      0.1525    2.991        5.566  9
```

```
##                      logLik  AICc delta weight
## 250 meter buffer   -119.272 260.2  0.00  0.332
## 4750 meter buffer  -119.970 261.6  1.40  0.165
## 4500 meter buffer  -120.043 261.8  1.54  0.153
## 5000 meter buffer  -120.409 262.5  2.28  0.106
## 4250 meter buffer  -120.499 262.7  2.45  0.097
## 3500 meter buffer  -121.314 264.3  4.08  0.043
## 3750 meter buffer  -121.371 264.4  4.20  0.041
## 4000 meter buffer  -121.492 264.7  4.44  0.036
## 3250 meter buffer  -122.235 266.1  5.93  0.017
## 3000 meter buffer  -123.530 268.7  8.52  0.005
## 2750 meter buffer  -124.189 270.1  9.84  0.002
## 2250 meter buffer  -125.196 272.1 11.85  0.001
## 2500 meter buffer  -125.314 272.3 12.08  0.001
## 2000 meter buffer  -126.005 273.7 13.47  0.000
## 1750 meter buffer  -127.337 276.3 16.13  0.000
## 500 meter buffer   -127.578 276.8 16.61  0.000
## 1250 meter buffer  -127.889 277.5 17.23  0.000
## 1000 meter buffer  -127.951 277.6 17.36  0.000
## 1500 meter buffer  -128.580 278.8 18.62  0.000
## 750 meter buffer   -129.879 281.4 21.22  0.000
## Models ranked by AICc(x)
```

> The 250m buffer comes out on top - similar to OSM scale results (although this wasn't the case when I first ran it with all variables regardless of correlation, it was 1250 I think)

Also of note the model results look exactly the same for both the glm and glmmTMB function- just wanted to test this since we've used both in the past.

This code chunk below we can use if we want to plot the different buffer scores later, but not running for now

```
# run model selection and save the results as a tibble for graphing use later
black_bear_global_model.sel <- model.sel(black_bear_buff) %>%

  as.data.frame() %>%

  rownames_to_column(var = 'Model')  %>%

  mutate(Dataset = deparse(substitute(prop_det_data)))

# look at model selection results
black_bear_global_model.sel
```

250m models

Now that we have best fit buffer size (250m) we can run several candidate models based on a priori hypotheses for what covariates might influence monthly presence/absence of black bears in our study area.

```
# Null model
 bbear_null <- glm(cbind(black_bear, absent_black_bear) ~ 1,
          data = prop_det_data$`250 meter buffer`,
          family = 'binomial')
```

```r
# global model
bbear_global <- glm(cbind(black_bear, absent_black_bear) ~
                      harvest +
                      roads +
                      pipeline +
                      seismic_lines +
                      wells +
                      lc_agriculture +
                      lc_broadleaf  +
                      lc_grassland  +
                      lc_mixed  +
                      lc_shrub,
                    data = prop_det_data$`250 meter buffer`,
                    family = 'binomial')

# Natural heterogeneity
bbear_nat <- glm(cbind(black_bear, absent_black_bear) ~
                      lc_broadleaf  +
                      lc_grassland  +
                      lc_mixed  +
                      lc_shrub,
                    data = prop_det_data$`250 meter buffer`,
                    family = 'binomial')

# forest harvest (TBD how this will look with yearly harvest data)
bbear_harvest <- glm(cbind(black_bear, absent_black_bear) ~
                      harvest,
                    data = prop_det_data$`250 meter buffer`,
                    family = 'binomial')

# transportation (roads) * at 250m buffer can combine with other models but cautious of pipeline r = 0.
bbear_rds <- glm(cbind(black_bear, absent_black_bear) ~
                      roads,
                    data = prop_det_data$`250 meter buffer`,
                    family = 'binomial')

# linear energy development
bbear_linear_energy <- glm(cbind(black_bear, absent_black_bear) ~
                      pipeline +
                      seismic_lines,
                    data = prop_det_data$`250 meter buffer`,
                    family = 'binomial')

# polygonal energy development
bbear_poly_energy <- glm(cbind(black_bear, absent_black_bear) ~
                      wells,
                    data = prop_det_data$`250 meter buffer`,
                    family = 'binomial')


# energy development
bbear_energy <- glm(cbind(black_bear, absent_black_bear) ~
                      pipeline +
```

```
                      seismic_lines +
                      wells,
                    data = prop_det_data$`250 meter buffer`,
                    family = 'binomial')


# polygonal disturbance (harvest + polygonal energy development + agriculture)
bbear_poly <- glm(cbind(black_bear, absent_black_bear) ~
                      harvest +
                      wells +
                      lc_agriculture,
                    data = prop_det_data$`250 meter buffer`,
                    family = 'binomial')


# linear disturbance (transportation + linear energy development)
bbear_linear <- glm(cbind(black_bear, absent_black_bear) ~
                      roads +
                      pipeline +
                      seismic_lines,
                    data = prop_det_data$`250 meter buffer`,
                    family = 'binomial')


# natural cover (natural heterogeneity + harvest + agriculture)
bbear_cover <- glm(cbind(black_bear, absent_black_bear) ~
                      harvest +
                      lc_agriculture +
                      lc_broadleaf  +
                      lc_grassland  +
                      lc_mixed  +
                      lc_shrub,
                    data = prop_det_data$`250 meter buffer`,
                    family = 'binomial')

# overall human disturbance (not applicable for all buffer sizes but this works for 250m)
bbear_disturb <- glm(cbind(black_bear, absent_black_bear) ~
                       harvest +
                       roads +
                       pipeline +
                       seismic_lines +
                       wells,
                     data = prop_det_data$`250 meter buffer`,
                     family = 'binomial')
```

Model selection

```
# compare black bear models
model.sel(bbear_null,
          bbear_global,
          bbear_nat,
```

```
          bbear_harvest,
          bbear_rds,
          bbear_linear_energy,
          bbear_poly_energy,
          bbear_energy,
          bbear_poly,
          bbear_linear,
          bbear_cover,
          bbear_disturb)
```

```
## Model selection table
##                       (Int)      hrv  lc_agr lc_brd lc_grs  lc_mxd lc_shr    ppl
## bbear_linear        -1.374                                                  3.495
## bbear_energy        -1.382                                                 -1.337
## bbear_linear_energy -1.428                                                 -2.667
## bbear_disturb       -1.219 -0.32610                                         4.633
## bbear_poly_energy   -1.540
## bbear_global        -1.913 -0.80270 0.03185 0.8064  1.916  0.1946  1.569 -2.672
## bbear_nat           -2.248                  0.6128  1.603 -0.2170  1.022
## bbear_rds           -1.547
## bbear_null          -1.647
## bbear_poly          -1.549  0.02589 0.04641
## bbear_harvest       -1.704  0.19340
## bbear_cover         -2.354 -0.30330 1.60300 0.7525  1.807 -0.1160  1.471
##                         rds ssm_lns    wll df   logLik  AICc delta weight
## bbear_linear        -22.710  -79.46         4 -123.868 256.5  0.00  0.356
## bbear_energy                -70.12 -10.76  4 -124.275 257.3  0.81  0.237
## bbear_linear_energy         -77.11         3 -125.708 257.9  1.37  0.179
## bbear_disturb       -23.650  -75.91 -10.10  6 -122.304 258.2  1.75  0.148
## bbear_poly_energy                  -14.14  2 -128.686 261.6  5.11  0.028
## bbear_global         -7.736  -69.25 -11.98 11 -117.387 262.4  5.91  0.018
## bbear_nat                                  5 -126.031 263.2  6.72  0.012
## bbear_rds           -16.530                2 -129.719 263.7  7.17  0.010
## bbear_null                                 1 -131.465 265.0  8.52  0.005
## bbear_poly                        -14.01   4 -128.683 266.1  9.63  0.003
## bbear_harvest                              2 -131.277 266.8 10.29  0.002
## bbear_cover                                7 -125.412 267.0 10.54  0.002
## Models ranked by AICc(x)
```

Top model/s summary

```
# linear
summary(bbear_linear)
```

```
##
## Call:
## glm(formula = cbind(black_bear, absent_black_bear) ~ roads +
##     pipeline + seismic_lines, family = "binomial", data = prop_det_data$`250 meter buffer`)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
```

```
## -2.5130  -1.5416  -0.5229   0.7790   3.0304
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -1.3743     0.1235 -11.133  < 2e-16 ***
## roads         -22.7127    12.0108  -1.891  0.05862 .
## pipeline        3.4952     4.6337   0.754  0.45066
## seismic_lines -79.4633    27.1450  -2.927  0.00342 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 162.6  on 58  degrees of freedom
## Residual deviance: 147.4  on 55  degrees of freedom
##   (1 observation deleted due to missingness)
## AIC: 255.74
##
## Number of Fisher Scoring iterations: 5
```

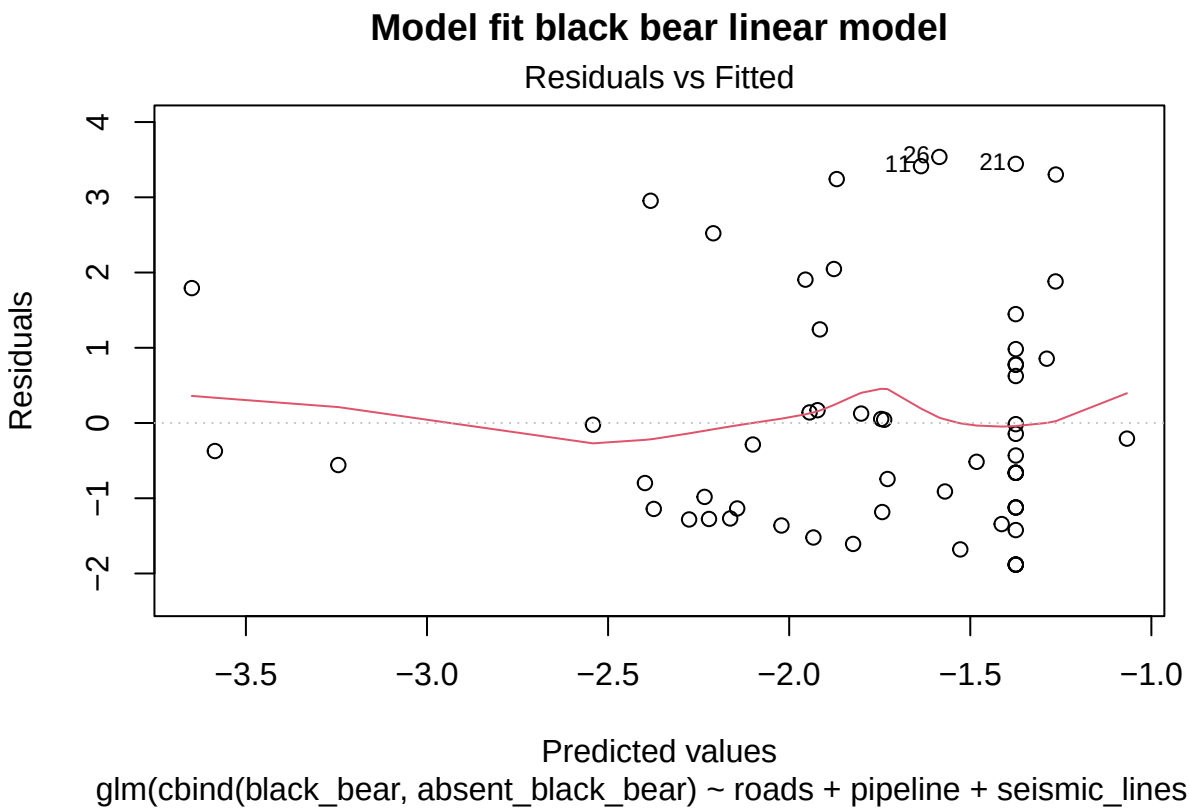
Model fit

We will start with VIF

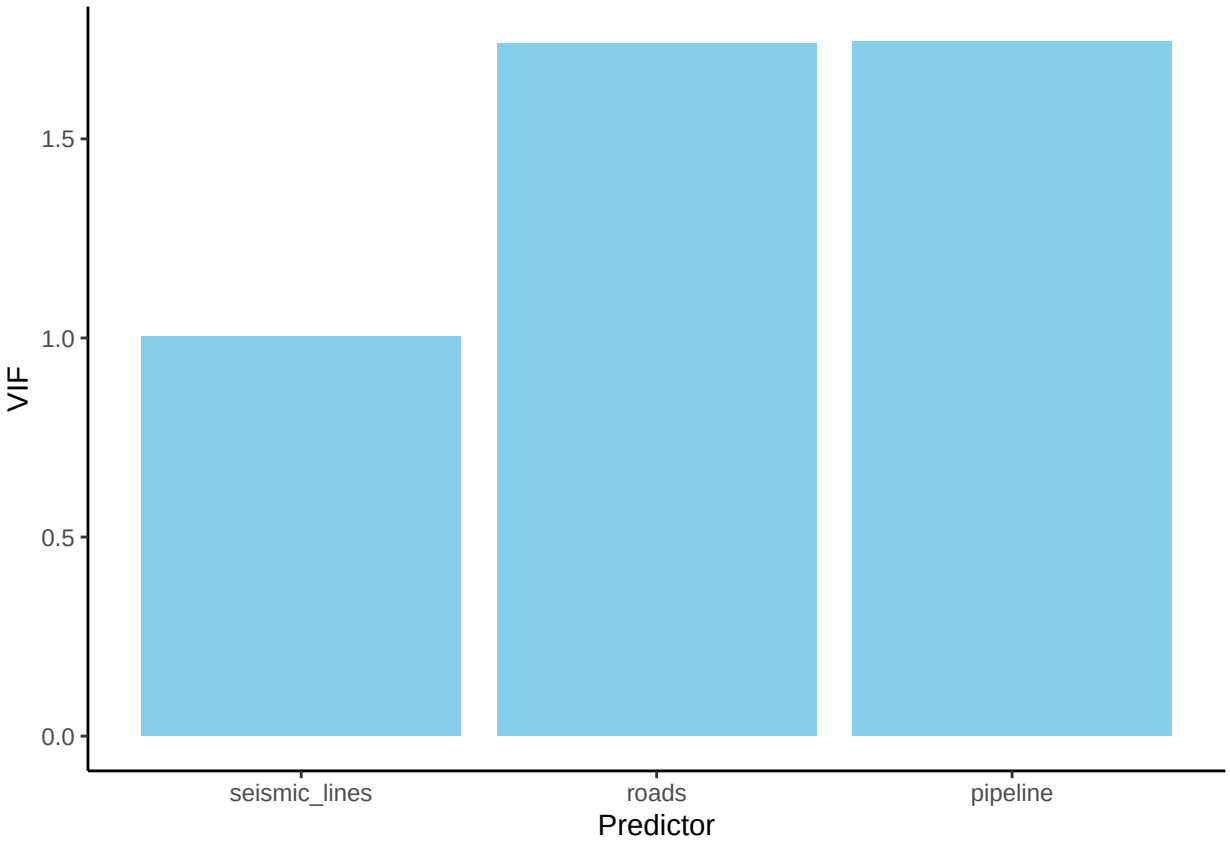
```
vif(bbear_linear)
```

```
##         roads      pipeline seismic_lines
##      1.739757      1.744948      1.004056
```

```
  plot(bbear_linear,
       which = 1,
       main = 'Model fit black bear linear model')
```




```
# calculate vif
vif(bbear_linear) %>%

  # Converts the named vector returned by vif() into a tidy tibble
  enframe(name = 'Predictor',
          value = 'VIF') %>%

  # plot with ggplot
  ggplot(aes(x = reorder(Predictor, VIF), # reorders from smallest VIF to largest (not sure I want like
             y = VIF)) +

  # plot as bars
  geom_bar(stat = 'identity', fill = 'skyblue') +

  # add labels
  labs(x = 'Predictor',
       y = 'VIF') +

  # set theme
  theme_classic()
```


VIF
1.5
1.0
0.5
0.0
seismic_lines
roads
pipeline
Predictor