Classifying Forest Cover

Contents

Description

Using 13 temporal and geographic variables, the goal of this project is to classify the type of tree cover in the Roosevelt National Forest of northern Colorado, just outside of Fort Collins. The forest consists of 6 wilderness areas that have, according to the University of California, Irvine's project depository, experienced minimal human disturbances, leaving the tree cover to be the result of natural processes.

The Kaggle Competition for which this data was made available asks four questions:

1. Can you build a model that predicts what types of trees grow in an area based on the surrounding characteristics?
2. What kinds of trees are most common in the Roosevelt National Forest?
3. Which tree types can grow in more diverse environments?
4. Are there certain tree types that are sensitive to an environmental factor, such as elevation or soil type?

The 13 variables included in the dataset are:

- **Cover_Type**: One of seven types of tree cover found in the forest. In the data downloaded for the project, the observations are coded 1:7. We have renamed them for clarity. Observations are based on the primary cover type of 30m x 30m areas, as determined by the United States Forest Service. This is our response variable.
- **Wilderness_Area**: Of the six wilderness areas in the Roosevelt National Forest, four were used in this dataset. In the original dataset, these were one-hot encoded. We put them in long form for better visualisation and because most machine learning methods will automatically one-hot encode categorical variables for us.
- **Soil_Type**: 40 soil types were identified in the dataset and more detailed information regarding the types can be found at https://www.kaggle.com/uciml/forest-cover-type-dataset. Similar to `Wilderness_Area`, `Soil_Type` was originally one-hot encoded.
- **Elevation**: The elevation of the observation in meters above sea level.
- **Aspect**: The aspect of the observation in degrees azimuth.
- **Slope**: The slope at which the observation is observed in degrees.

- **Hillshade_9am**: The amount of hillshade for the observation at 09:00 on the summer solstice. This is a value between 0 and 225.
- **Hillshade_Noon**: The amount of hillshade for the observation at 12:00 on the summer solstice. This is a value between 0 and 225.
- **Hillshade_3pm**: The amount of hillshade for the observation at 15:00 on the summer solstice. This is a value between 0 and 225.
- **Vertical_Distance_To_Hydrology**: Vertical distance to nearest water source in meters. Negative numbers indicate distance below a water source.
- **Horizontal_Distance_To_Hydrology**: Horizontal distance to nearest water source in meters.
- **Horizontal_Distance_To_Roadways**: Horizontal distance to nearest roadway in meters.
- **Horizontal_Distance_To_Fire_Points**: Horizontal distance to nearest wildfire ignition point in meters.

Outline

1. We are going to begin by examining the structure of the data and manipulating it to serve our visualisation needs. This will include:
    - Checking for missing values and duplicates.
    - Converting the data into a tidy structure by putting the types of soil and wilderness areas into long form and converting categorical areas to factors.
    - Giving names to the `Wilderness_Area` and `Cover_Type` variables to replace the indicator values.
    - Reordering the data for convenience and clarity.
2. We will then visualise the data to give us a better idea of what variables might be good predictor variables.
3. Following visualisation, we will determine which predictor variables to use by building a random forest model with all variables and choosing only those with a high enough importance.
4. We will then split the data with an 80/20 train/test split.
5. Three machine learning models will be built with a 10-fold cross validation for each. All models will be built using the caret package:
    - kNN - k nearest neighbours from the `knn` package
    - random forest using:
        - the rf method from the `randomforest` package
        - the ranger method from the `ranger` package
6. Lastly, we will test the models using our test dataset to make sure the models generalise well.

Packages

```
library(tidyverse) # for data exploration and manipulation
library(skimr) # for useful and beautiful summary statistics
library(corrplot) # for correlation testing
library(caret) # for machine learning
library(reticulate) # for running python code
```

Data Exploration

Data Summary

Let's take a look at structure of the data.

```
cov <- read_csv('covType.csv')
glimpse(cov)
```

```
Observations: 581,012
Variables: 55
$ Elevation                           <dbl> 2596, 2590, 2804, 2785, 259...
$ Aspect                              <dbl> 51, 56, 139, 155, 45, 132, ...
$ Slope                               <dbl> 3, 2, 9, 18, 2, 6, 7, 4, 9,...
$ Horizontal_Distance_To_Hydrology    <dbl> 258, 212, 268, 242, 153, 30...
$ Vertical_Distance_To_Hydrology      <dbl> 0, -6, 65, 118, -1, -15, 5,...
$ Horizontal_Distance_To_Roadways     <dbl> 510, 390, 3180, 3090, 391, ...
$ Hillshade_9am                       <dbl> 221, 220, 234, 238, 220, 23...
$ Hillshade_Noon                      <dbl> 232, 235, 238, 238, 234, 23...
$ Hillshade_3pm                       <dbl> 148, 151, 135, 122, 150, 14...
$ Horizontal_Distance_To_Fire_Points <dbl> 6279, 6225, 6121, 6211, 617...
$ Wilderness_Area1                    <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
$ Wilderness_Area2                    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Wilderness_Area3                    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Wilderness_Area4                    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type1                          <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type2                          <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type3                          <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type4                          <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type5                          <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type6                          <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type7                          <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type8                          <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type9                          <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type10                         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type11                         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type12                         <dbl> 0, 0, 1, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type13                         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type14                         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type15                         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type16                         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type17                         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type18                         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type19                         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type20                         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type21                         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type22                         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type23                         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type24                         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type25                         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type26                         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type27                         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type28                         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type29                         <dbl> 1, 1, 0, 0, 1, 1, 1, 1, 1, ...
```

```
$ Soil_Type30                           <dbl> 0, 0, 0, 1, 0, 0, 0, 0, 0, ...
$ Soil_Type31                           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type32                           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type33                           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type34                           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type35                           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type36                           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type37                           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type38                           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type39                           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Soil_Type40                           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ Cover_Type                            <dbl> 5, 5, 2, 2, 5, 2, 5, 5, 5, ...
```

We should also check for duplicated rows and missing values before we go any further.

```
# test for duplicates; approximately 13.5 times faster than using base::duplicated
nrow(cov) - nrow(distinct(cov))
```

```
0
```

```
skim_to_wide(cov)
```

```
# A tibble: 55 x 13
   type  variable missing complete n     mean  sd    p0    p25   p50
   <chr> <chr>    <chr>   <chr>    <chr> <chr> <chr> <chr> <chr> <chr>
 1 nume~ Aspect   0       581012   5810~ "  1~ " 11~ 0     58    127
 2 nume~ Cover_T~ 0       581012   5810~ "   ~ "   ~ 1     1     2
 3 nume~ Elevati~ 0       581012   5810~ " 29~ " 27~ 1859  2809  2996
 4 nume~ Hillsha~ 0       581012   5810~ "  1~ "  3~ 0     119   143
 5 nume~ Hillsha~ 0       581012   5810~ "  2~ "  2~ 0     198   218
 6 nume~ Hillsha~ 0       581012   5810~ "  2~ "  1~ 0     213   226
 7 nume~ Horizon~ 0       581012   5810~ " 19~ "132~ 0     1024  1710
 8 nume~ Horizon~ 0       581012   5810~ "  2~ " 21~ 0     108   218
 9 nume~ Horizon~ 0       581012   5810~ " 23~ "155~ 0     1106  1997
10 nume~ Slope    0       581012   5810~ "   ~ "   ~ 0     9     13
# ... with 45 more rows, and 3 more variables: p75 <chr>, p100 <chr>,
#   hist <chr>
```

Let's take a moment to familiarise ourselves with a summary of the data. Using the oft overlooked `skimr` package for beautiful summary statistics, we can see that this is a massive dataset with 581,012 observations and 55 variables, but we are lucky enough to not have any missing data or duplicates. **TODO: MORE INFO ABOUT SUMMARY STATS**

Since we are only using the dataset as a learning tool, we will take a small sample so that our visualisations and machine learning calls can run efficiently. As we will be trying to predict `Cover_Type` using the variables in the dataset, we will take an equal number of samples from each of the seven types of cover to avoid class imbalances.

Additionally, we see that the `Soil_Type*` and `Wilderness_Area*` variables can be gathered into a tidy format for better visualisation. The `Cover_Type` variable is the variable of interest, so we are going to move that to the front of the data for convenience. We are also going to rename the values in the `Cover_Type` and `Wilderness_Area` variables to show the actual name of the value, rather than an indicator value. The names can be found on the Kaggle competition page. Because they are categorical variables, we should convert the `Cover_Type`, `Soil_Type` and `Wilderness_Area` variables to factors.

We will leave the `Hillshade_*` values in wide form for the moment.

```r
set.seed(1808)
cov_Tidy <- (cov %>%

  # take a subset of the overall data containing 1000 samples of each cover type
  group_by(Cover_Type) %>%
  sample_n(size = 1000) %>%
  ungroup() %>%

  # tidy Wilderness_Area* data and rename values
  gather(Wilderness_Area, Wilderness_Value, Wilderness_Area1:Wilderness_Area4) %>%
  filter(Wilderness_Value >= 1) %>%
  select(-Wilderness_Value) %>%
  mutate(Wilderness_Area = str_extract(Wilderness_Area, '\\d+'),
         Wilderness_Area = case_when(Wilderness_Area == 1 ~ 'Rawah',
                                     Wilderness_Area == 2 ~ 'Neota',
                                     Wilderness_Area == 3 ~ 'Comanche Peak',
                                     Wilderness_Area == 4 ~ 'Cache la Poudre'),
         Wilderness_Area = as.factor(Wilderness_Area)) %>%

  # tidy Soil_Type* data
  gather(Soil_Type, Soil_Value, Soil_Type1:Soil_Type40) %>%
  filter(Soil_Value == 1) %>%
  select(-Soil_Value) %>%
  mutate(Soil_Type = as.factor(str_extract(Soil_Type, '\\d+'))) %>%

  # rename Cover_Type variables
  mutate(Cover_Type = case_when(Cover_Type == 1 ~ 'Spruce/Fir',
                                Cover_Type == 2 ~ 'Lodgepole Pine',
                                Cover_Type == 3 ~ 'Ponderosa Pine',
                                Cover_Type == 4 ~ 'Cottonwood/Willow',
                                Cover_Type == 5 ~ 'Aspen',
                                Cover_Type == 6 ~ 'Douglas Fir',
                                Cover_Type == 7 ~ 'Krummholz'),
         Cover_Type = as.factor(Cover_Type)) %>%

  # reorder columns for convenience
  select(Cover_Type:Soil_Type,
         Elevation:Slope,
         Hillshade_9am:Hillshade_3pm,
         Vertical_Distance_To_Hydrology,
         Horizontal_Distance_To_Hydrology:Horizontal_Distance_To_Fire_Points))

glimpse(cov_Tidy)
```

```
Observations: 7,000
Variables: 13
$ Cover_Type                         <fct> Ponderosa Pine, Ponderosa P...
$ Wilderness_Area                    <fct> Cache la Poudre, Cache la P...
$ Soil_Type                          <fct> 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
$ Elevation                          <dbl> 2060, 2034, 2098, 2388, 217...
$ Aspect                             <dbl> 101, 48, 76, 92, 228, 98, 2...
$ Slope                              <dbl> 33, 27, 26, 20, 26, 23, 34,...
$ Hillshade_9am                      <dbl> 253, 219, 242, 246, 172, 25...
$ Hillshade_Noon                     <dbl> 178, 171, 181, 206, 253, 20...
```

```
$ Hillshade_3pm                      <dbl> 26, 71, 53, 80, 208, 66, 24...
$ Vertical_Distance_To_Hydrology     <dbl> 113, 105, 46, 32, 132, 25, ...
$ Horizontal_Distance_To_Hydrology   <dbl> 228, 162, 108, 201, 309, 10...
$ Horizontal_Distance_To_Roadways    <dbl> 268, 175, 1045, 612, 350, 3...
$ Horizontal_Distance_To_Fire_Points <dbl> 524, 458, 459, 636, 914, 92...
```

Now that we have a nice, tidy tibble, let's take another look at some summary statistics regarding our sample before we move on to crafting some visualisations.

```
skim_tee(cov_Tidy)
```

```
Skim summary statistics
 n obs: 7000
 n variables: 13

-- Variable type:factor ---------------------------------------------------------------------------
        variable missing complete    n n_unique
      Cover_Type       0     7000 7000        7
       Soil_Type       0     7000 7000       37
 Wilderness_Area       0     7000 7000        4
                                top_counts ordered
 Asp: 1000, Cot: 1000, Dou: 1000, Kru: 1000   FALSE
           10: 916, 29: 611, 3: 465, 4: 405   FALSE
  Com: 2972, Cac: 2168, Raw: 1621, Neo: 239   FALSE

-- Variable type:numeric --------------------------------------------------------------------------
                           variable missing complete    n    mean      sd
                             Aspect       0     7000 7000  157.11  109.68
                          Elevation       0     7000 7000 2746.79  417.78
                      Hillshade_3pm       0     7000 7000  135.09   46.18
                      Hillshade_9am       0     7000 7000  212.69   30.82
                     Hillshade_Noon       0     7000 7000  219.14   22.77
 Horizontal_Distance_To_Fire_Points       0     7000 7000 1519.54 1108.06
   Horizontal_Distance_To_Hydrology       0     7000 7000  227.78  207.57
    Horizontal_Distance_To_Roadways       0     7000 7000 1719.55 1333.8
                              Slope       0     7000 7000   16.59    8.5
     Vertical_Distance_To_Hydrology       0     7000 7000   51.51   60.91
   p0  p25    p50     p75 p100     hist
    0   66  126    261     359
 1872 2375 2752   3108    3840
    0  106  138    167     248
    0  196  220    236     254
  103  207  222    235     254
    0  751 1276   1974    6990
    0   67  180    324    1310
    0  757 1307.5 2295.25 6678
    0   10   16     22      52
 -119    6   32     81     597
```

A few take-aways from the summary statistics of our sample set:

- Our tidy data contains 1,000 samples of each `Cover_Type`. The original data was highly uneven in terms of this variable with Spruce/Fir and Lodgepole Pine types dominating the landscape.
- The `Wilderness_Area` numbers are much closer now as well, with the exception of the Neota region. This was the smallest of regions in terms of observations in the original data, but it was much closer to the Cache la Poudre region than it is following our sampling.

- The numerical values in the tidy data are not centred or scaled. This is great for us when it comes to visualising the data, but as we plan on running a kNN model, we will need to remember to pre-process the data before running the model.

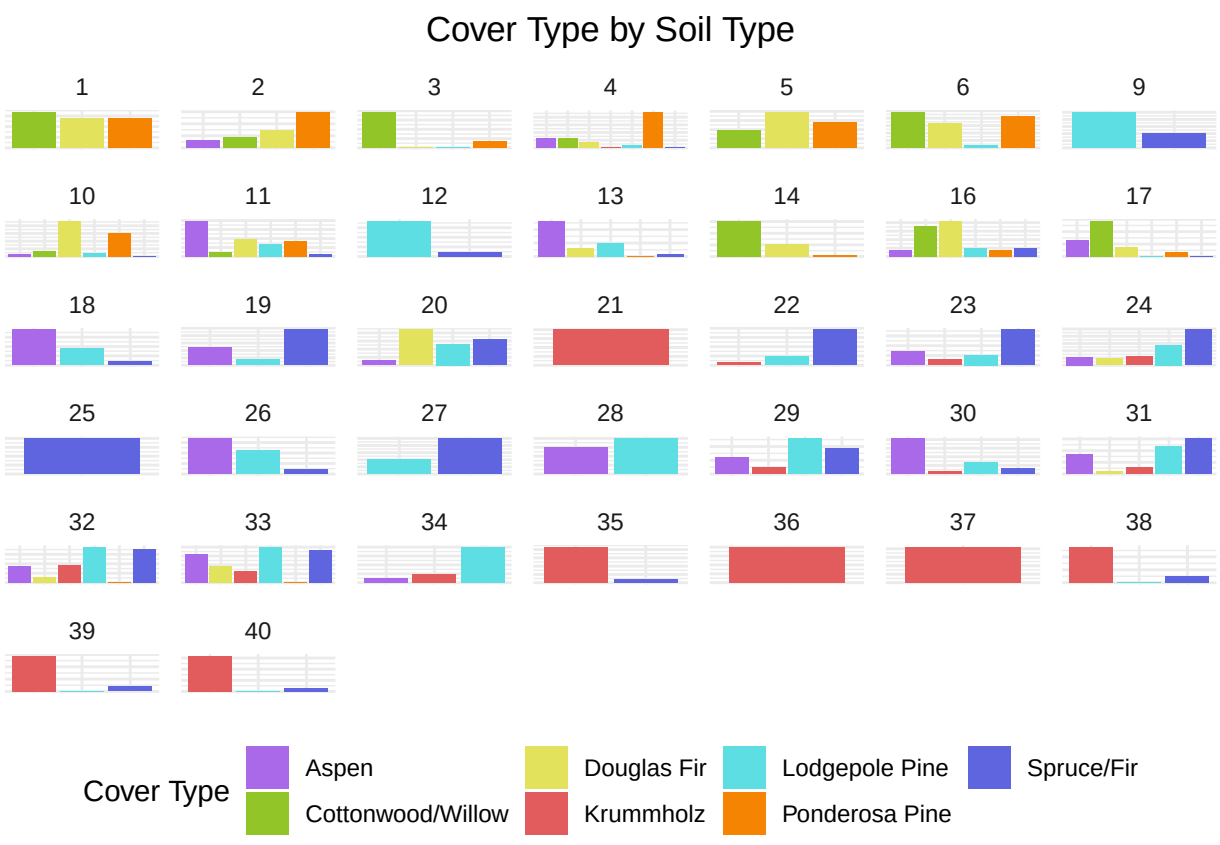
Data Visualisation

Soil Type

A good starting point is to see which type of cover grows in which type of soil. Different soil porosity and acidity are known to play a large role in determining what plants can grow in a particular area, and so `Soil_Type` is sure to be a good predictor variable. For the purpose of exploration, we aren't interested in specific values, but rather the variation of `Cover_Type` by Soil_Type.

```
palette <- c('#A969E9', '#92C628', '#E2E25C',
             '#E25C5E', '#5CDEE2', '#F58403', '#5F65DE')

ggplot(cov_Tidy, aes(x = Cover_Type, fill = Cover_Type)) +
  geom_bar() +
  facet_wrap(~reorder(Soil_Type, sort(as.integer(Soil_Type))), scales = 'free') +
  labs(fill = 'Cover Type', title = 'Cover Type by Soil Type') +
  scale_fill_manual(values = palette) +
  theme_minimal() +
  theme(legend.position = 'bottom',
        plot.title = element_text(hjust = 0.5),
        axis.title = element_blank(),
        axis.text = element_blank())
```

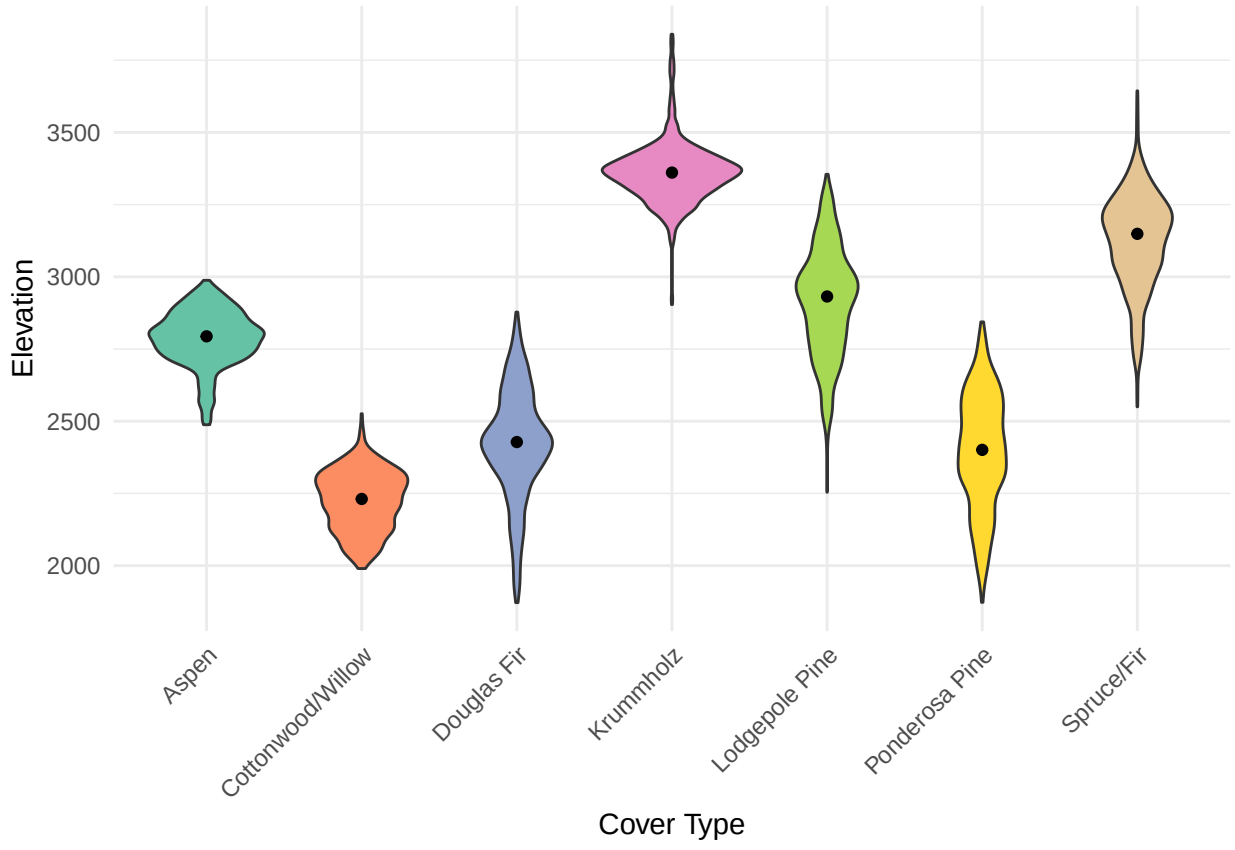


`Soil_Type`s 21, 25, 36 and 37 contain only a single cover type while others, such as 11 or 24, contain multiple. One thing to note is that there are several soil types where the only cover type is Krummholz. This is a type of cover that grows in harsh conditions, such as sub-alpine regions, where other plants are rare and soils may be poor in organic nutrients. Another point of interest is that Krummholz and Cottonwood/Willow covers do not share any soil types. You may also have noticed that soil types 7, 8 and 15 are missing; we simply did not have any instances of these types in our sample. Let's look at the role of `Elevation` and see if this explains any of our findings.

Elevation

Using violin plots (a combination of box plots and density plots), we can get a decent idea of how each cover type is distributed across our predictor variables.

```
ggplot(cov_Tidy, aes(x = Cover_Type, y = Elevation, fill = Cover_Type)) +
  geom_violin() +
  stat_summary(fun.y = 'median', geom = 'point') +
  labs(x = 'Cover Type') +
  scale_fill_brewer(palette = 'Set2') +
  theme_minimal() +
  theme(legend.position = 'none',
        axis.text.x = element_text(angle = 45,
                                   hjust = 1))
```

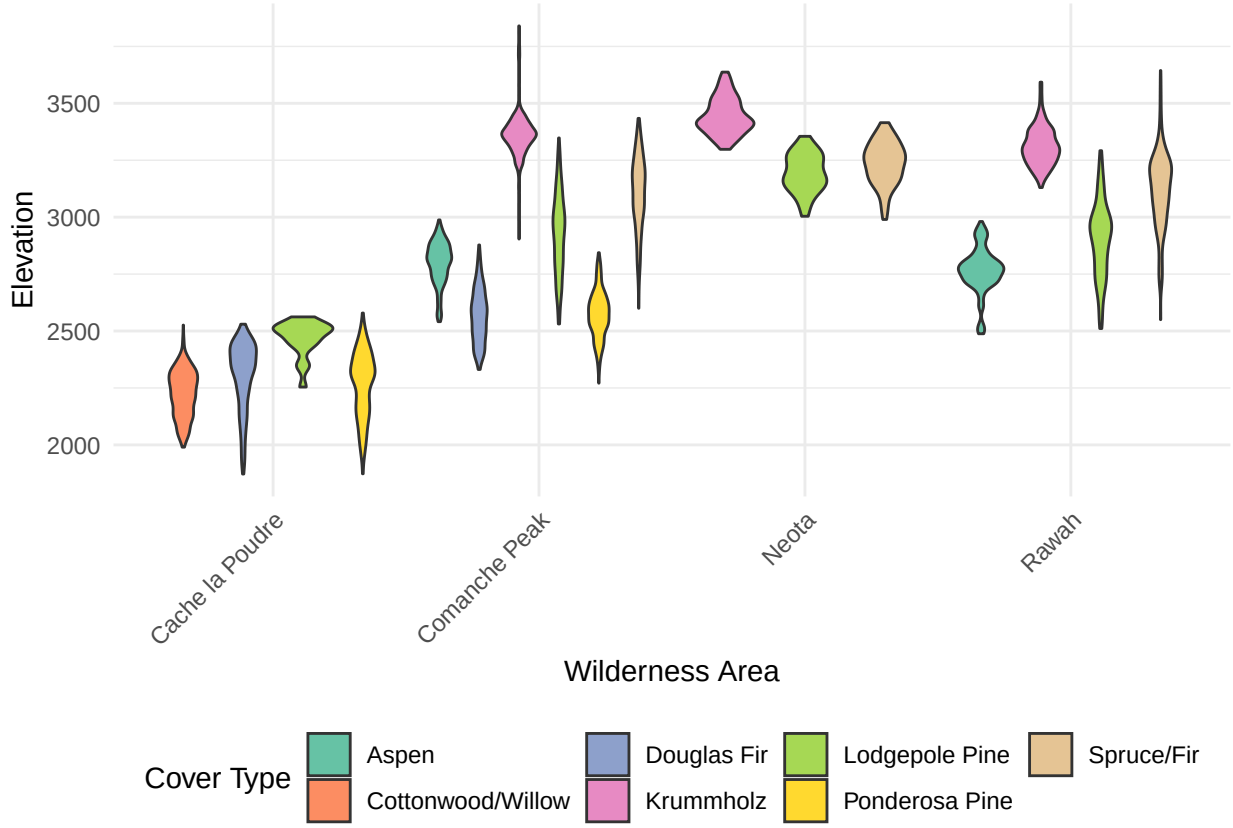


As expected, Krummholz cover is found at very high elevations where other plants are rare. Not only do Krummholz and Cottonwood/Willow covers not share any of the same soil types, but they aren't found within nearly 500 meters of elevation to each other.

The UCI dataset page shows that the Neota wilderness area has the highest mean elevation, followed closely

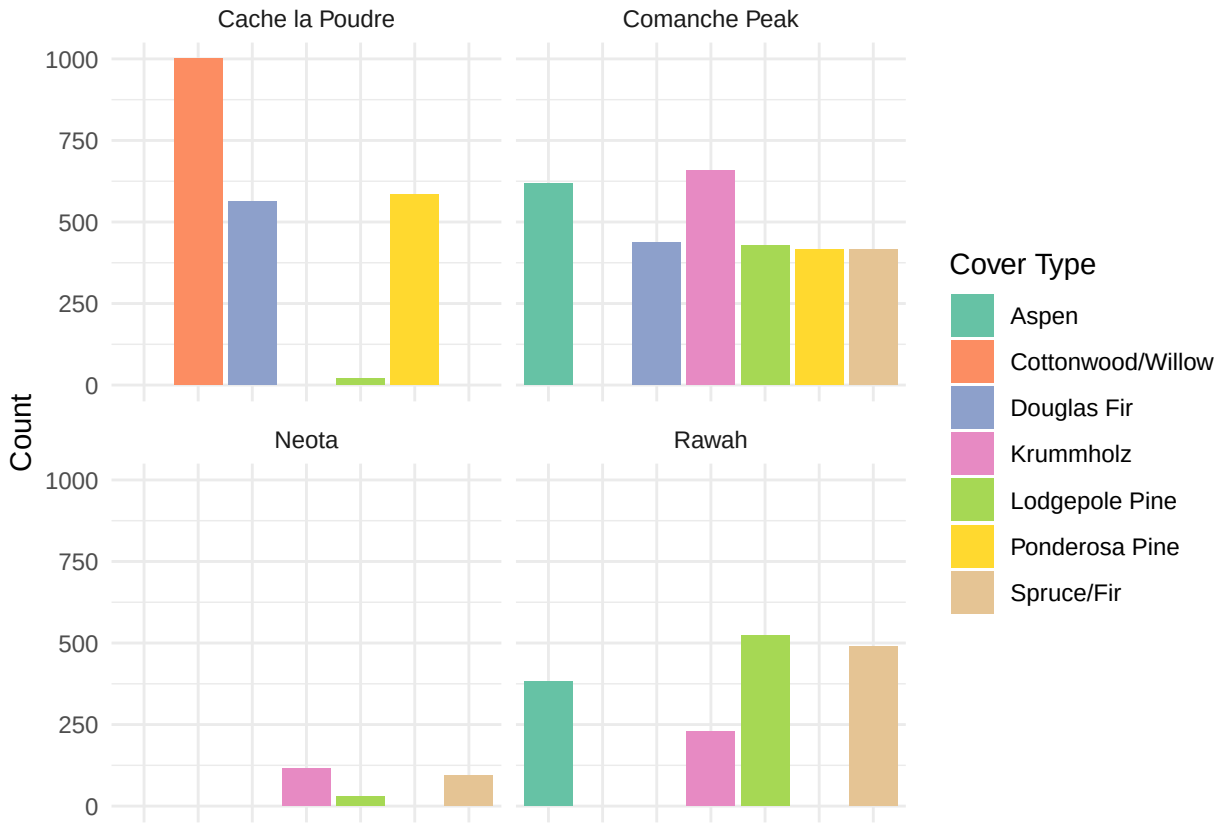
by the Rawah and Comanche Peak areas. Looking at a map of the Roosevelt National Forest tells us that the Cache la Poudre area is the area surrounding the lower-lying Cache la Poudre river. With our previous figure showing that `Elevation` plays a role in determining cover type, we should expect to find a different composition of `Cover_Type` in this area.

```
ggplot(cov_Tidy, aes(x = Wilderness_Area, y = Elevation, fill = Cover_Type)) +
  geom_violin() +
  labs(x = 'Wilderness Area') +
  scale_fill_brewer(name = 'Cover Type',
                    palette = 'Set2') +
  theme_minimal() +
  theme(legend.position = 'bottom',
        axis.text.x = element_text(angle = 45,
                                   hjust = 1))
```



Keeping in mind that we are only working with a sample of the overall data, we can see that Cottonwood/Willow trees are only found in the Cache la Poudre area and Krummholz trees are only found in the upper elevations of the other three areas. A bar plot gives a better look at the distribution of `Cover_Type` by `Wilderness_Area`.

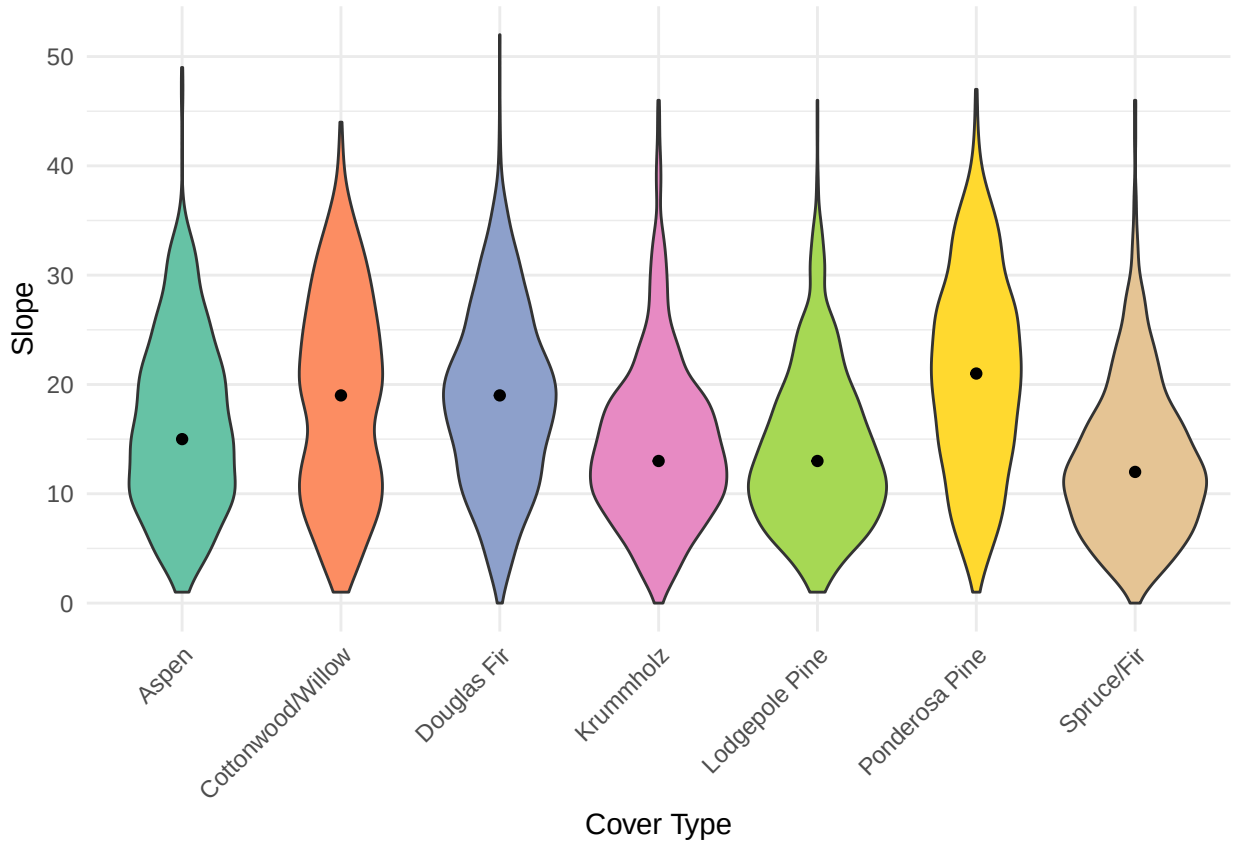
```
ggplot(cov_Tidy, aes(x = Cover_Type, fill = Cover_Type)) +
  geom_bar() +
  facet_wrap(~Wilderness_Area) +
  labs(y = 'Count') +
  scale_fill_brewer(name = 'Cover Type',
                    palette = 'Set2') +
  theme_minimal() +
  theme(axis.text.x = element_blank(),
        axis.title.x = element_blank())
```



Slope

`Slope` is another geographic variable that might play a role in determining the type of cover in an area. Steeper slopes likely feature higher water run-off and increased erosion, leading to the need for cover types with stronger root systems that might be more tolerant to drier conditions.

```
ggplot(cov_Tidy, aes(x = Cover_Type, y = Slope, fill = Cover_Type)) +
  geom_violin() +
  stat_summary(fun.y = 'median', geom = 'point') +
  labs(x = 'Cover Type') +
  scale_fill_brewer(palette = 'Set2') +
  theme_minimal() +
  theme(legend.position = 'none',
        axis.text.x = element_text(angle = 45,
                                   hjust = 1))
```



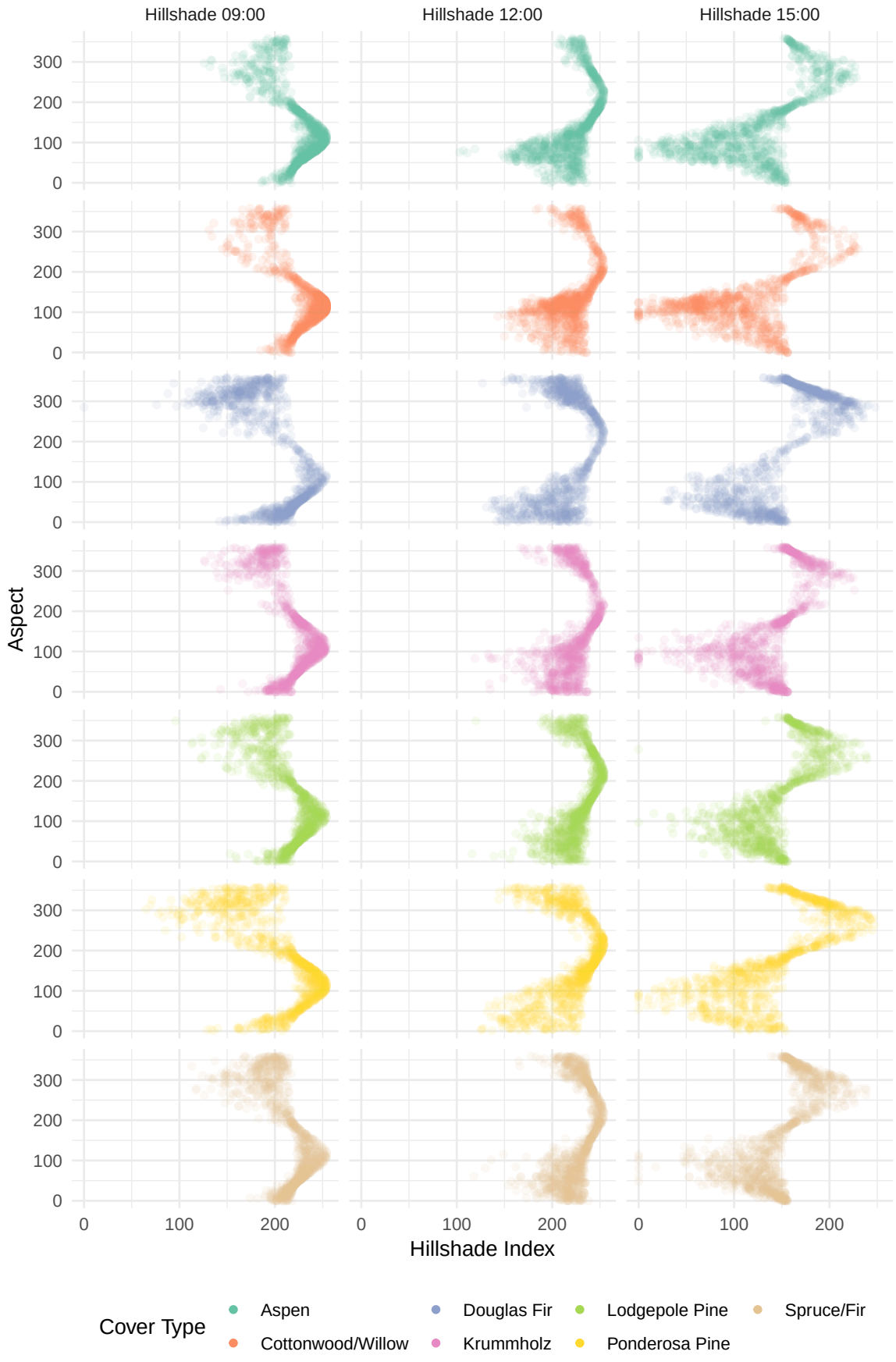
Within our sample, there doesn't seem to be much variation in terms of `Slope` when measured across `Cover_Type`. At most, there appears to be an 8 or 9 degree difference between the medians of the Ponderosa Pine and the Spruce/Fir types, but the ranges are highly similar for all covers. We can expect that slope may play a role, but it will not likely be as important as `Soil_Type` or `Elevation`.

Aspect and Hillshade

The dataset also contains some temporal data in the form of hillshade index values measured throughout the day. These values can be plotted against the aspect values to give us an idea of how sunlight, direction and time influence `Cover_Type`.

```
cov_Tidy %>%
  gather(Hillshade, Hillshade_Index, Hillshade_9am:Hillshade_3pm) %>%
  mutate(Hillshade = factor(Hillshade, levels = c('Hillshade_9am',
                                                  'Hillshade_Noon',
                                                  'Hillshade_3pm'),
                            labels = c('Hillshade 09:00',
                                       'Hillshade 12:00',
                                       'Hillshade 15:00'))) %>%
  ggplot(aes(x = Hillshade_Index, y = Aspect, colour = Cover_Type)) +
  geom_point(alpha = 0.1) +
  facet_grid(Cover_Type ~ Hillshade) +
  labs(x = 'Hillshade Index') +
  scale_colour_brewer(name = 'Cover Type',
                      palette = 'Set2') +
  guides(colour = guide_legend(override.aes = list(alpha = 1))) +
  theme_minimal() +
```

```
  theme(legend.position = 'bottom',
        strip.text.y = element_blank())
```


Hillshade 09:00
Hillshade 12:00
Hillshade 15:00
Aspect
Hillshade Index
0
100
200
300
Cover Type
Aspen
Cottonwood/Willow
Douglas Fir
Krummholz
Lodgepole Pine
Ponderosa Pine
Spruce/Fir

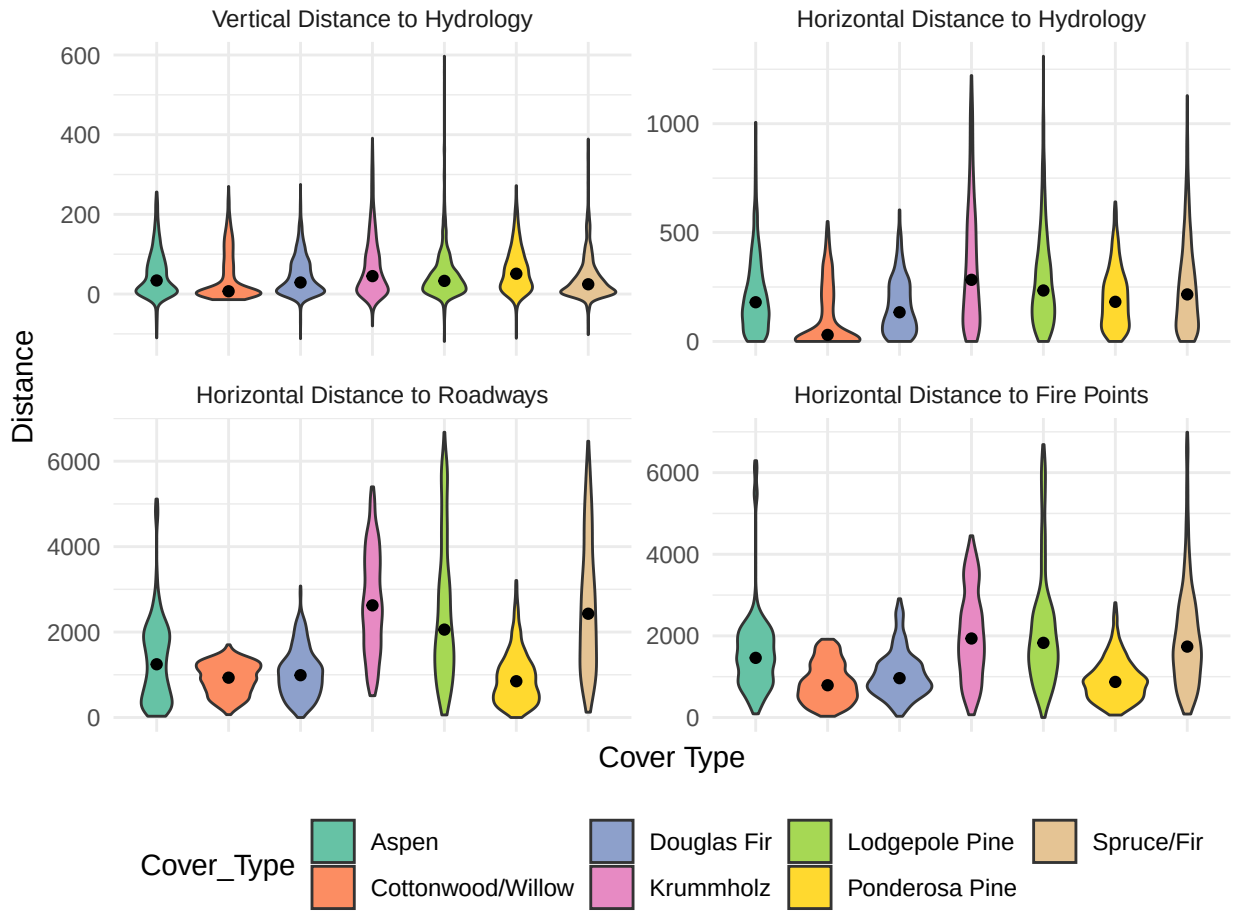
Shade Index ~ Aspect figures can be difficult to read because of the shear number of points on the plot; each facet in the above graph contains 1000 points, many of which are overlapping. When we reduce the alpha levels so that the points are nearly transparent (changing the shape to something hollow could also be useful here) and view the figure on a large enough screen, some interesting patterns start to emerge. For example, Cottonwood/Willow cover mostly has aspects between 100 and 150 degrees azimuth, roughly facing east-southeast, creating a large amount of shade during the morning and much less during the afternoon.

Distance to Landmarks

The dataset contains four measures of distance to three type of landmarks: water sources, roadways and fire points.

- Distance to water sources could show the extent of a cover's dependence on water. Lands closer to a water source are likely to have soil which is more saturated than those far away.
- Distance to roadways might indicate how well a type of cover deals with human interference; conversely, it could just show that roadways are built in a certain area, i.e. in lower elevations.
- Distance to fire points can be seen as a metric of a cover type's ability to regrow following a forest fire; types of cover which can regrow quickly will take hold of a new area long before a slow-growing cover.

```
cov_Tidy %>%
  gather(Measure, Distance,
         Vertical_Distance_To_Hydrology:Horizontal_Distance_To_Fire_Points) %>%
  mutate(Measure = factor(Measure,
                          levels = c('Vertical_Distance_To_Hydrology',
                                     'Horizontal_Distance_To_Hydrology',
                                     'Horizontal_Distance_To_Roadways',
                                     'Horizontal_Distance_To_Fire_Points'),
                          labels = c('Vertical Distance to Hydrology',
                                     'Horizontal Distance to Hydrology',
                                     'Horizontal Distance to Roadways',
                                     'Horizontal Distance to Fire Points'))) %>%
  ggplot(aes(x = Cover_Type, y = Distance, fill = Cover_Type)) +
  geom_violin() +
  stat_summary(fun.y = 'median', geom = 'point',
               show.legend = FALSE) +
  facet_wrap(~Measure, scales = 'free_y') +
  labs(x = 'Cover Type') +
  scale_fill_brewer(palette = 'Set2') +
  theme_minimal() +
  theme(legend.position = 'bottom',
        axis.text.x = element_blank())
```

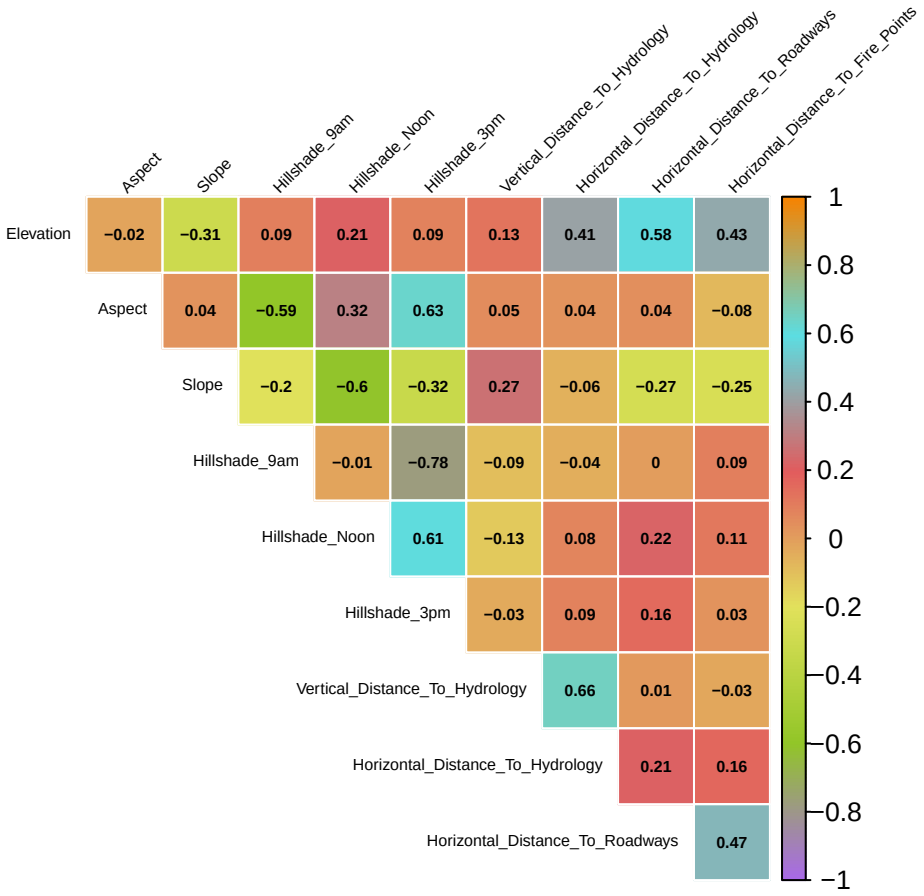


With the exception of `Vertical_Distance_To_Hydrology`, there seems to be a good amount of variation between `Cover_Types` for the distance measurements. `Vertical_Distance_To_Hydrology` is on the smallest scale by about 800 meters, and all of the medians are within about 50 meters of each other. There is some variation in the ranges for this measurement, but the other three metrics might be better predictors of `Cover_Type`.

Modelling

We should always consider correlations in the data. The `corrplot` function provides a succinct graph to visualise correlation between variables. Correlations are not necessarily bad, but we should be aware of them because they can impact certain machine learning models.

```
col <- colorRampPalette(c('#A969E9', '#92C628', '#E2E25C',
                          '#E25C5E', '#5CDEE2', '#F58403'))
corrplot(cor(cov_Tidy[4:13]), method = 'color', type = 'upper',
         diag = FALSE, addgrid.col = 'white', col = col(200),
         tl.cex = 0.5, tl.srt = 45, tl.col = 'black',
         addCoef.col = 'black', number.cex = 0.5)
```



For the most part, the data in our sample is uncorrelated with a few exceptions:

- `Hillshade_9am` and `Hillshade_3pm` are negatively correlated and `Hillshade_Noon` and `Hillshade_3pm` are positively correlated. Interestingly, `Hillshade_9am` and `Hillshade_Noon` show virtually no correlation (-0.01).
- `Slope` and `Hillshade_Noon` are positively correlated.
- `Aspect` is also correlated with `Hillshade_9am` (negatively) and `Hillshade_3pm` (positively), which we would expect.
- Vertical and horizontal distances to hydrology are positively correlated.

Due to the nature of random variable selection and single variable splits with random forest models, correlated values aren't much of a concern. There is always a chance that the correlated values won't be chosen together for each decision, provided the mtry value is not too high and, even if the correlated values are chosen together, the splits are based only on one predictor variable.

Correlated values can pose a problem for kNN models, but we can deal with this by performing feature selection prior to building a kNN model. There are two ways we can do this:

1. by running a PCA
2. by building a random forest model with all variables and using the built-in importance measures.

Variable Selection

For variable selection using a random forest model, we will use the `caret` package and the 'rf' method from the `randomForest` package. We will perform a 10-fold cross-validation to test out-of-sample performance.

We set the `importance` argument to `TRUE` and then use the `varImp` function on the output to provide us with a data frame listing the relative importance of each variable on predicting the `Cover_Type`.

We use our tidied data as input into our random forest model because the model will automatically one-hot encode all of the categorical variables.

```
vars <- train(Cover_Type ~ ., data = cov_Tidy,
              method = 'rf',
              importance = TRUE,
              trControl = trainControl(method = 'cv',
                                       number = 3,
                                       verboseIter = TRUE))
varImp(vars)
```

Based on the importance results, we are going to go ahead and use all the variables to build our models. We could reduce the number used by removing some individual soil types or wilderness areas, but we wouldn't have any real justification for doing so in this case.

Model Building

Let's start building some models by splitting our data into training and test sets. We will use an 80/20 split.

```
set.seed(1808)
cov_Tidy_Train <- sample_frac(cov_Tidy, 0.8)
cov_Tidy_Test <- anti_join(cov_Tidy, cov_Tidy_Train)
```

We can also build our trainControl now to save some work later. We are going to use a simple 10-fold cross-validation to test for out-of-sample performance.

```
control <- trainControl(method = 'cv', number = 3)
```

K Nearest Neighbours

For our k-Nearest Neighbors model, we need to remember to pre-process the data by centring and scaling it. We can also select a range of values of k for our model to test. Here, we will use the square root of the number of rows of our training data +/- 10.

```
set.seed(1808)
mod_knn <- train(Cover_Type ~ ., data = cov_Tidy_Train,
                 method = 'knn',
                 tuneGrid = expand.grid(k = (round(sqrt(nrow(cov_Tidy) * 0.8)) - 10):
                                        (round(sqrt(nrow(cov_Tidy) * 0.8)) + 10)),
                 preProc = c('center', 'scale'),
                 trControl = control)
mod_knn
preds_knn <- predict(mod_knn, cov_Tidy_Test)
mean(preds_knn == cov_Tidy_Test$Cover_Type)
```

Random Forest with rf

```
set.seed(1808)
mod_rf <- train(Cover_Type ~ ., data = cov_Tidy_Train,
                method = 'rf',
                trControl = control)
```

```
mod_rf
preds_rf <- predict(mod_rf, cov_Tidy_Test)
mean(preds_rf == cov_Tidy_Test$Cover_Type)
```

Random Forest with ranger

```
set.seed(1808)
mod_ranger <- train(Cover_Type ~ ., data = cov_Tidy_Train,
                    method = 'ranger',
                    trControl = control)
mod_ranger
preds <- predict(mod_ranger, cov_Tidy_Test)
mean(preds == cov_Tidy_Test$Cover_Type)
```