Capstone Data Exploration

Tom Thorpe

April 17, 2018

Objective

View different plots of the cleaned Forest Cover data from the previous section to learn more about the data.

Include required libraries.

```
progStart=Sys.time()
print(paste("R script started at",progStart))
```

```
## [1] "R script started at 2018-06-07 06:27:25"
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(ggridges) # for easier viewing of sub-group distributions
```

Point to data. The forestcover_clean_full.csv is the cleaned data to be graphed.

```
infile="C:/Users/Tom/git/datasciencefoundation/ForestCoverage/forestcover_clean_full.csv"
#infile="C:/Users/Tom/git/datasciencefoundation/ForestCoverage/forestcover_clean.csv"
#infile="C:/Users/Tom/git/datasciencefoundation/ForestCoverage/forestcoversmall_clean_full.csv"
#infile="C:/Users/Tom/git/datasciencefoundation/ForestCoverage/forestcoversmall_clean.csv"
out2file="C:/Users/Tom/git/datasciencefoundation/ForestCoverage/forestcover_graph.csv"
#out1file="C:/Users/Tom/git/datasciencefoundation/ForestCoverage/forestcoversmall_clean_full.csv"
#out2file="C:/Users/Tom/git/datasciencefoundation/ForestCoverage/forestcoversmall_clean.csv"

alphaVal<-0.05 # large data
#alphaVal<-0.1  # small data
```

Load the data.

```
startTime=Sys.time()
print(paste("Data load started at",startTime))
```

```
## [1] "Data load started at 2018-06-07 06:27:27"
```

```
forestcover <- read.csv(infile,header=TRUE,sep=",") %>% tbl_df()
```

endTime=Sys.time() print(paste(“Data load completed at”,endTime)) print(paste(“Elapsed time=”,endTime-startTime,“seconds.”)) “‘

Data Overview

The forest cover data has a row for each sample representing a 30 meter by 30 meter square area of land. Each cell sample is described by elevation, slope and direction the cell faces, distance to water, roads, and fire and binary columns for wilderness area and soil type. One of 4 possible wilderness areas and one of 40 possible aggregated soil types are set in each row. The predicted variable is the coverage type indicating 1 of 7 possible trees found in the cell sample.

The data is described in detail here: https://archive.ics.uci.edu/ml/machine-learning-databases/covtype/covtype.info. The data names have been abbreviated but can be related to the data descriptions easily.

```
glimpse(forestcover)
```

```
## Observations: 581,012
## Variables: 131
## $ Elev                  <int> 2596, 2590, 2804, 2785, 2595, 2579, 26...
## $ Aspect                <int> 51, 56, 139, 155, 45, 132, 45, 49, 45,...
## $ Slope                 <int> 3, 2, 9, 18, 2, 6, 7, 4, 9, 10, 4, 11,...
## $ H2OHD                 <int> 258, 212, 268, 242, 153, 300, 270, 234...
## $ H2OVD                 <int> 0, -6, 65, 118, -1, -15, 5, 7, 56, 11,...
## $ RoadHD                <int> 510, 390, 3180, 3090, 391, 67, 633, 57...
## $ Shade9AM              <int> 221, 220, 234, 238, 220, 230, 222, 222...
## $ Shade12PM             <int> 232, 235, 238, 238, 234, 237, 225, 230...
## $ Shade3PM              <int> 148, 151, 135, 122, 150, 140, 138, 144...
## $ FirePtHD              <int> 6279, 6225, 6121, 6211, 6172, 6031, 62...
## $ RWwild                <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ NEwild                <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ CMwild                <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ CPwild                <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST01                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST02                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST03                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST04                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST05                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST06                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST07                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST08                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST09                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST10                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST11                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST12                  <int> 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST13                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST14                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST15                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST16                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST17                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST18                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,...
## $ ST19                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST20                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST21                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST22                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST23                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST24                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST25                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST26                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
```

```
## $ ST27                    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST28                    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST29                    <int> 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0,...
## $ ST30                    <int> 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1,...
## $ ST31                    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST32                    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST33                    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST34                    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST35                    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST36                    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST37                    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST38                    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST39                    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ST40                    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ CovType                 <int> 5, 5, 2, 2, 5, 2, 5, 5, 5, 5, 5, 2, 2,...
## $ STsum                   <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ Wildsum                 <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ Wilderness_Area         <fct> Rawah, Rawah, Rawah, Rawah, Rawah, Raw...
## $ ClimateName             <fct> SubAlpine, SubAlpine, Montane, SubAlpi...
## $ GeoName                 <fct> Ign_Meta, Ign_Meta, Ign_Meta, Ign_Meta...
## $ CovName                 <fct> Aspen, Aspen, Lodgepole, Lodgepole, As...
## $ ElevSlot                <int> 25, 25, 28, 27, 25, 25, 26, 26, 26, 26...
## $ SoilType                <int> 29, 29, 12, 30, 29, 29, 29, 29, 29, 29...
## $ ClimateZone             <int> 7, 7, 4, 7, 7, 7, 7, 7, 7, 7, 6, 7, 7,...
## $ GeoZone                 <int> 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7,...
## $ Montane_low             <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Montane                 <int> 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,...
## $ SubAlpine               <int> 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ Alpine                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Dry                     <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Non_Dry                 <int> 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,...
## $ Alluvium                <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Glacial                 <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Sed_mix                 <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Ign_Meta                <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ Aquolis_cmplx           <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Argiborolis_Pachic      <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Borohemists_cmplx       <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Bross                   <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Bullwark                <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Bullwark_Cmplx          <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Catamount               <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Catamount_cmplx         <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Cathedral               <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Como                    <int> 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1,...
## $ Cryaquepts_cmplx        <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Cryaquepts_Typic        <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Cryaquolls              <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Cryaquolls_cmplx        <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Cryaquolls_Typic        <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Cryaquolls_Typic_cmplx  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Cryoborolis_cmplx       <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Cryorthents             <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Cryorthents_cmplx       <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
```

```
## $ Cryumbrepts              <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Cryumbrepts_cmplx        <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Gateview                 <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Gothic                   <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Granile                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Haploborolis             <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Legault                  <int> 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Legault_cmplx            <int> 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1,...
## $ Leighcan                 <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Leighcan_cmplx           <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Leighcan_warm            <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Moran                    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Ratake                   <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Ratake_cmplx             <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Rogert                   <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,...
## $ Supervisor_Limber_cmplx <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Troutville               <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Unspecified              <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Vanet                    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Wetmore                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Bouldery_ext             <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Rock_Land                <int> 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1,...
## $ Rock_Land_cmplx          <int> 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Rock_Outcrop             <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Rock_Outcrop_cmplx       <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Rubbly                   <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Stony                    <int> 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Stony_extreme            <int> 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1,...
## $ Stony_very               <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,...
## $ Till_Substratum          <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Spruce_Fir               <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ LodgepolePine            <int> 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1,...
## $ PonderosaPine            <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Cottonwood_Willow        <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Aspen                    <int> 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0,...
## $ DouglasFir               <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Krummholz                <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
```

An Alternate Forest Cover data frame is created for Shapiro statistics testing for whether data follows a normal distribution. The Shapiro test has a maximum of 5000 data points. The Forestcover data frame is sampled to ensure the maximum is not exceeded.

```
if(nrow(forestcover)> 5000) {
  samplefactor <- as.integer(nrow(forestcover)/4500)
  if (samplefactor < 2) { samplefactor <- 2 }
  altforestcover<-forestcover[seq(1, nrow(forestcover), samplefactor), ]
} else {
  altforestcover<-forestcover
}
```

List Selected Data Ranges

Data ranges are listed to help validate the data is within expected limits.

```
myranges <- function(name,x) { c(name, min = min(x), mean = mean(x), median=median(x), max = max(x),
                                 nonzero=sum(x!=0)) }

forestDataRanges <- data.frame("Data"=character(), "min"=double(), "mean"=double(),
                               "median"=double(), "max"=double(), "nonzero"=integer(),
                               stringsAsFactors=FALSE)
forestDataRanges[nrow(forestDataRanges)+1,] <- myranges("Elev",forestcover$Elev)
forestDataRanges[nrow(forestDataRanges)+1,] <- myranges("Elev",forestcover$ElevSlot)
forestDataRanges[nrow(forestDataRanges)+1,] <- myranges("Aspect",forestcover$Aspect)
forestDataRanges[nrow(forestDataRanges)+1,] <- myranges("Slope",forestcover$Slope)
forestDataRanges[nrow(forestDataRanges)+1,] <- myranges("H2OHD",forestcover$H2OHD)
forestDataRanges[nrow(forestDataRanges)+1,] <- myranges("H2OVD",forestcover$H2OVD)
forestDataRanges[nrow(forestDataRanges)+1,] <- myranges("RoadHD",forestcover$RoadHD)
forestDataRanges[nrow(forestDataRanges)+1,] <- myranges("FirePtHD",forestcover$FirePtHD)
forestDataRanges[nrow(forestDataRanges)+1,] <- myranges("Shade9AM",forestcover$Shade9AM)
forestDataRanges[nrow(forestDataRanges)+1,] <- myranges("Shade12P",forestcover$Shade12PM)
forestDataRanges[nrow(forestDataRanges)+1,] <- myranges("Shade3PM",forestcover$Shade3PM)
forestDataRanges[nrow(forestDataRanges)+1,] <- myranges("RWwild",forestcover$RWwild)
forestDataRanges[nrow(forestDataRanges)+1,] <- myranges("NEwild",forestcover$NEwild)
forestDataRanges[nrow(forestDataRanges)+1,] <- myranges("CMwild",forestcover$CMwild)
forestDataRanges[nrow(forestDataRanges)+1,] <- myranges("CPwild",forestcover$CPwild)
forestDataRanges
```

```
##        Data  min              mean median  max nonzero
## 1      Elev 1859  2959.36530054457   2996 3858  581012
## 2      Elev   18  29.0998344268277     29   38  581012
## 3    Aspect    0  155.656807432549    127  360  576098
## 4     Slope    0  14.1037035379648     13   66  580356
## 5     H2OHD    0  269.428216628916    218 1397  556409
## 6     H2OVD -173   46.418855376481     30  601  542347
## 7    RoadHD    0  2350.14661142971   1997 7117  580888
## 8  FirePtHD    0   1980.291226343   1710 7173  580961
## 9  Shade9AM    0  212.146048618617    218  254  580999
## 10 Shade12P    0  223.318716308785    226  254  581007
## 11 Shade3PM    0   142.52826275533    143  254  579674
## 12   RWwild    0 0.448865083681576      0    1  260796
## 13   NEwild    0 0.051434393781884      0    1   29884
## 14   CMwild    0 0.436073609495157      0    1  253364
## 15   CPwild    0 0.063626913041383      0    1   36968
```

The results show all the data values have reasonable values and there is no missing data. The elevation ranges from 1859 meters (6099 feet) to 3858 meters (12657 feet). These are valid ranges for elevation in the Colorado wilderness areas being sampled, but the rule of thumb for timberline (the maximum elevation for where trees are found) is 11500 feet. It might be interesting to see how accurate predictions are if samples above 11800 feet are removed.

ElevSlot, "Elevation Slot" is a new column that creates bins for elevation data for use with Chi-square testing. It is calculated by diving the elevation by 100 and truncating the value by saving as an integer. This results in 21 bins.

The Aspect which is the compass heading that the terrain faces, ranges from 0 to 360 degrees and is a valid data range. The Slope is the steepness of the terrain with 0 degrees being flat and 90 degrees being vertical. The maximum Slope was found to be 66 degrees which seems logical since trees are not usually seen on near-vertical cliffs. (It's a different story in New Zealand!)

The horizontal distance to the nearest water features, range from 0 to 1397 meters which seems reasonable.

The vertical distance to nearest water features, range from -173 to 601 meters which seems reasonable and can be negative since the nearest water may be below the forest cover data sample.

The horizontal distance to the nearest road ranges from 0 to 7117 meters which is reasonable. The horizontal distance to the nearest fire features range from 0 to 7173 meters which is reasonable. The amount of shade present in a cell sample at 9AM, 12PM and 3PM ranges from 0 (full sun) to 254 (fully shaded).

```
##print(paste("Forest coverage ST__ column deletion started at",startTime))
#forestcover <- forestcover %>% select(-ST01,-ST02,-ST03,-ST04,-ST05,-ST06,-ST07,-ST08,-ST09,-ST10,
#                                      -ST11,-ST12,-ST13,-ST14,-ST15,-ST16,-ST17,-ST18,-ST19,-ST20,
#                                      -ST21,-ST22,-ST23,-ST24,-ST25,-ST26,-ST27,-ST28,-ST29,-ST30,
#                                      -ST31,-ST32,-ST33,-ST34,-ST35,-ST36,-ST37,-ST38,-ST39,-ST40,
#                                      -STsum, -Wildsum
#                                      )

#forestcover <- mutate(forestcover,CovName = "")
#forestcover$CovName[forestcover$CovType == 1] <- "Spruce&Fir"
#forestcover$CovName[forestcover$CovType == 2] <- "Lodgepole"
#forestcover$CovName[forestcover$CovType == 3] <- "Ponderosa"
#forestcover$CovName[forestcover$CovType == 4] <- "Cotton&Willow"
#forestcover$CovName[forestcover$CovType == 5] <- "Aspen"
#forestcover$CovName[forestcover$CovType == 6] <- "DouglasFir"
#forestcover$CovName[forestcover$CovType == 7] <- "Krummholz"

#forestcover <- mutate(forestcover,Climate = "")
#forestcover$Climate[forestcover$Montane_low == 1 & forestcover$Non_Dry ==1] <- "1Montane_Low_NonDry"
#forestcover$Climate[forestcover$Montane == 1] <- "2Montane"
#forestcover$Climate[forestcover$Subalpine == 1] <- "3SubAlpine"
#forestcover$Climate[forestcover$Alpine == 1] <- "4Alpine"

#forestcover <- mutate(forestcover,Wilderness_Area = "")
#forestcover$Wilderness_Area[forestcover$RWwild == 1] <- "Rawah"
#forestcover$Wilderness_Area[forestcover$NEwild == 1] <- "Neota"
#forestcover$Wilderness_Area[forestcover$CMwild == 1] <- "Comanche"
#forestcover$Wilderness_Area[forestcover$CPwild == 1] <- "Cache"
#forestcover$Wilderness_Area <- as.factor(forestcover$Wilderness_Area)
```

Data distributions

Now check some basic distributions.

Elevation Frequency

```
jpeg(filename="Fig-Elev-Freq.jpg")
plot(table(forestcover$Elev),
  main="Elevation Frequency Chart", # sub="All Data",
  xlab="Elevation (meters)", ylab="count")
  # xlim=c(xmin, xmax), ylim=c(ymin, ymax)))
dev.off()
```

```
## pdf
##   2
```

An elevation frequency chart using the basic plot commands show the shape of the distribution of the elevation in meters.

Elevation Density Chart

```
g <- ggplot(forestcover,aes(Elev)) +
       geom_density()   +#
       labs(title = 'Elevation Density',
            subtitle = 'All Data',
            x="Elevation (meters)", y="Density")
       # theme(axis.text.x = element_text(angle=-90))
ggsave("Fig-Elev-Density.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

The elevation frequency looks reasonable for Colorado high country.

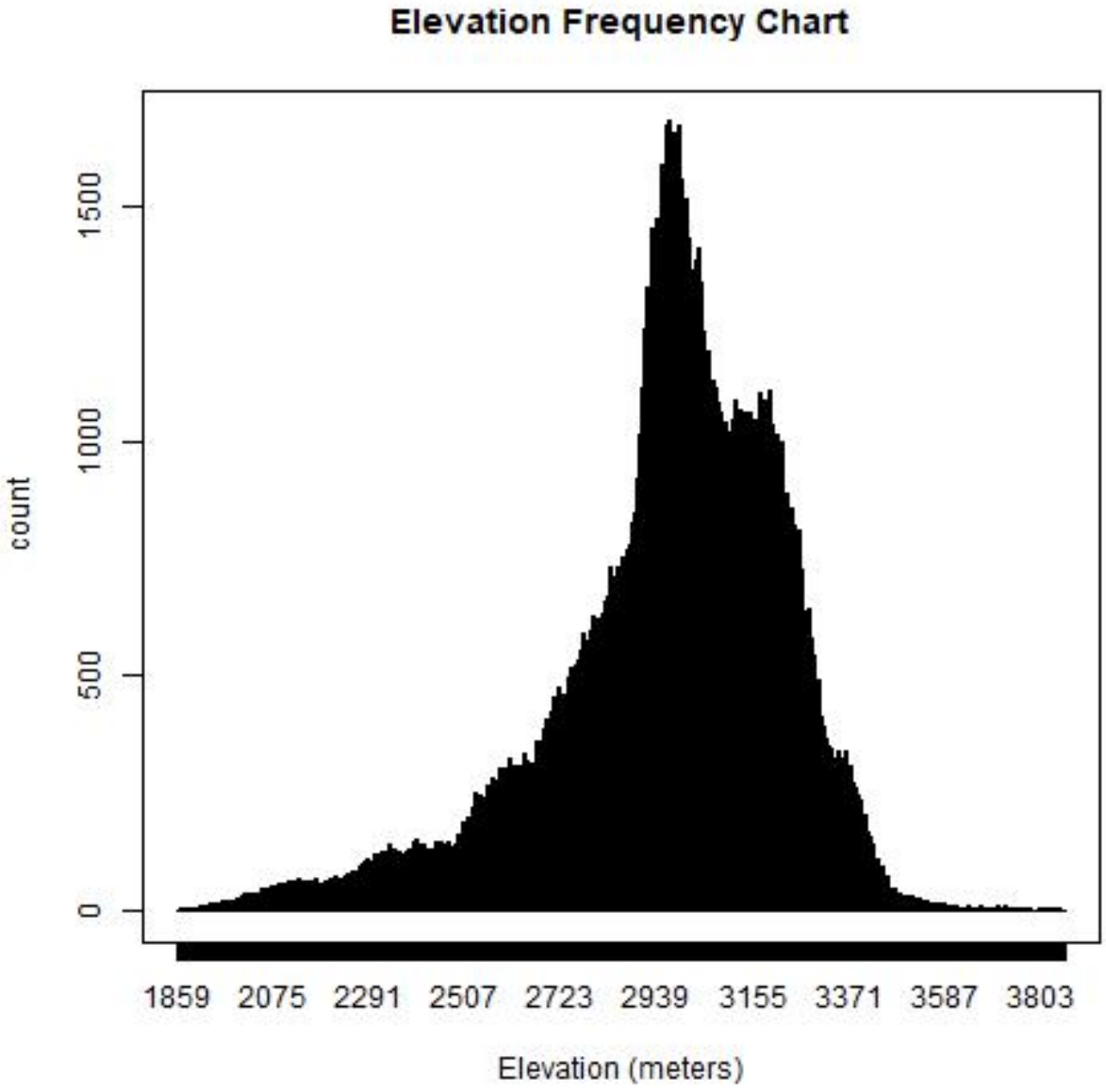

Figure 1: Elevation Frequency

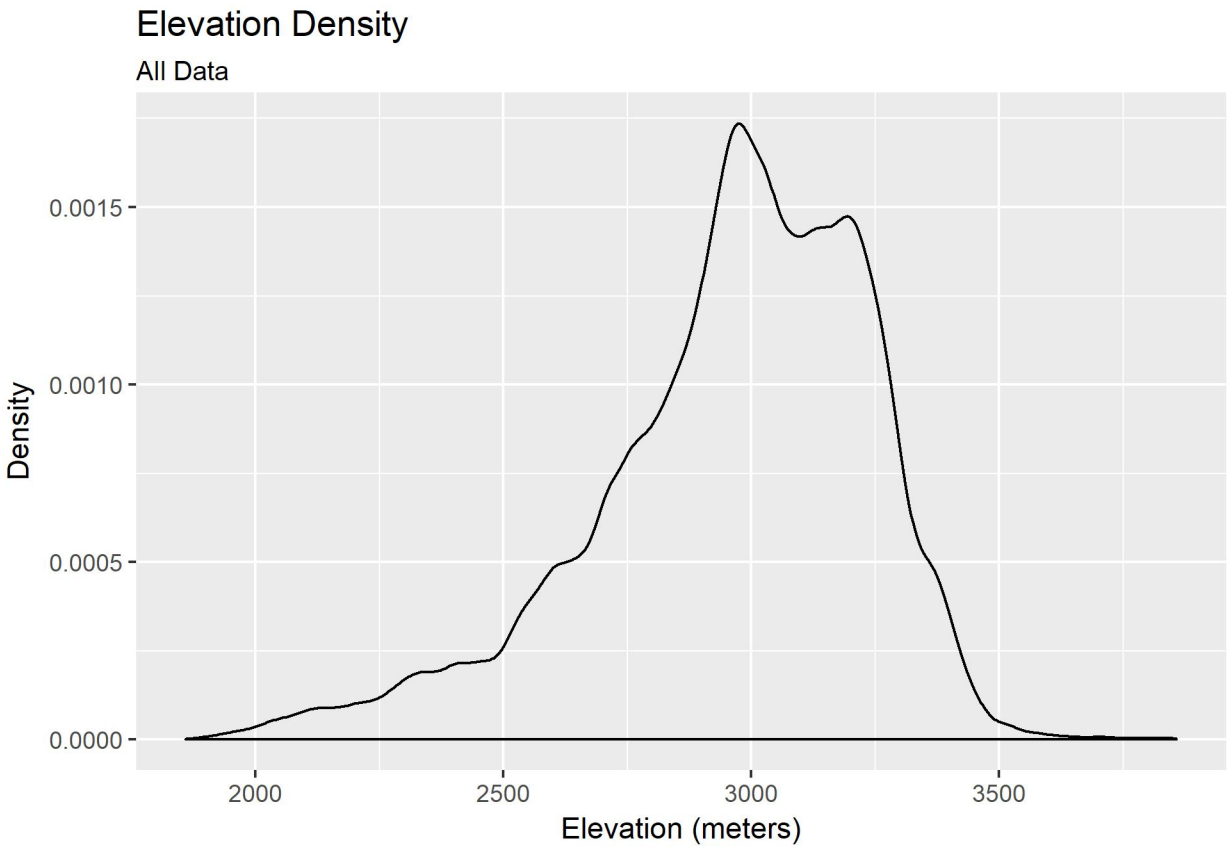

Figure 2: Elevation Density

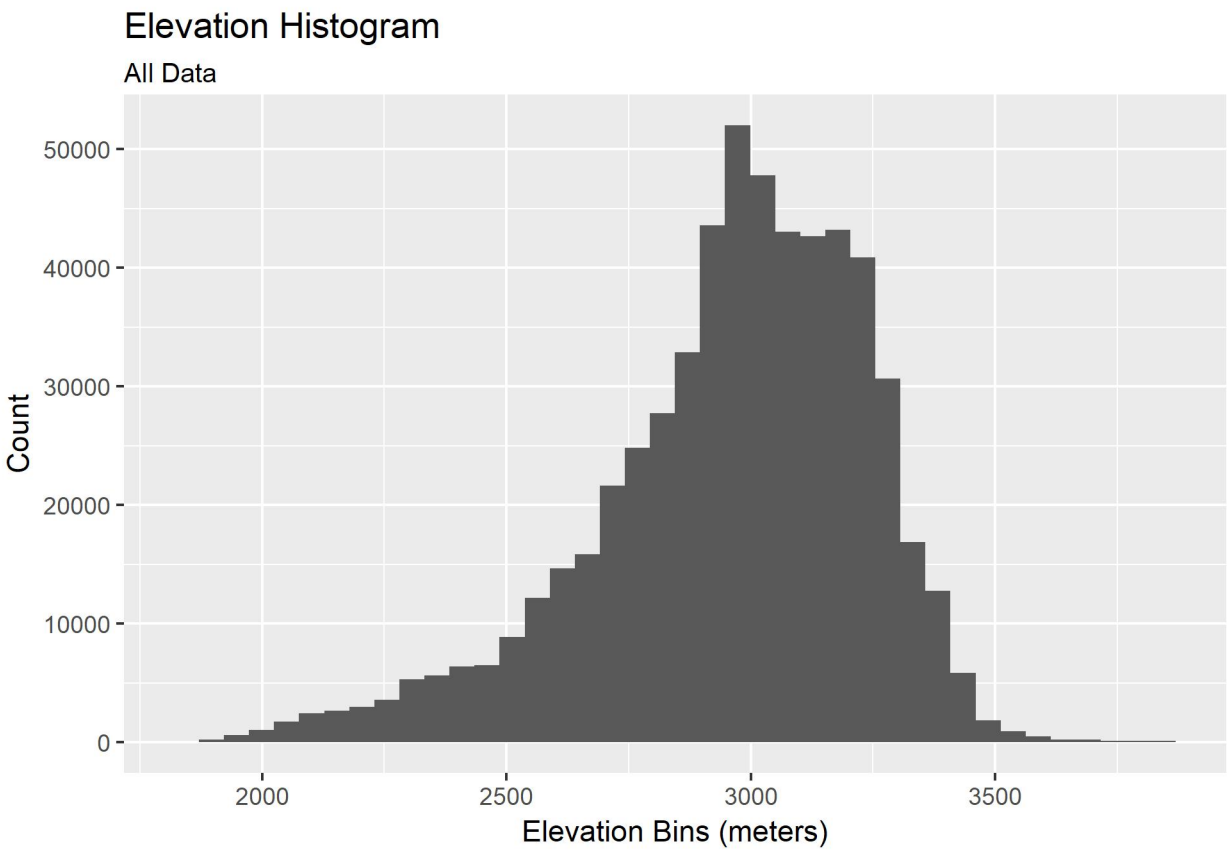

Figure 3: Elevation Histogram

Elevation Histogram

```
g <- ggplot(forestcover,aes(Elev)) +
     geom_histogram(bins=40) +
     labs(title = 'Elevation Histogram',
            subtitle = 'All Data',
            x="Elevation Bins (meters)", y="Count")
       # theme(axis.text.x = element_text(angle=-90))
ggsave("Fig-Elev-Histogram.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

A good histogram is generated Using 40 bins. There are two humps in the histogram and there may be a more complicated distribution. The elevation may be related to other variables. The elevation is grouped by coverage type and wilderness.

Elevation Density by Coverage Type

```
g <- ggplot(forestcover,aes(Elev, fill=CovName)) +
     geom_density(alpha=0.3) +
     labs(title = 'Elevation Density',
            subtitle = 'By Coverage Type',
```

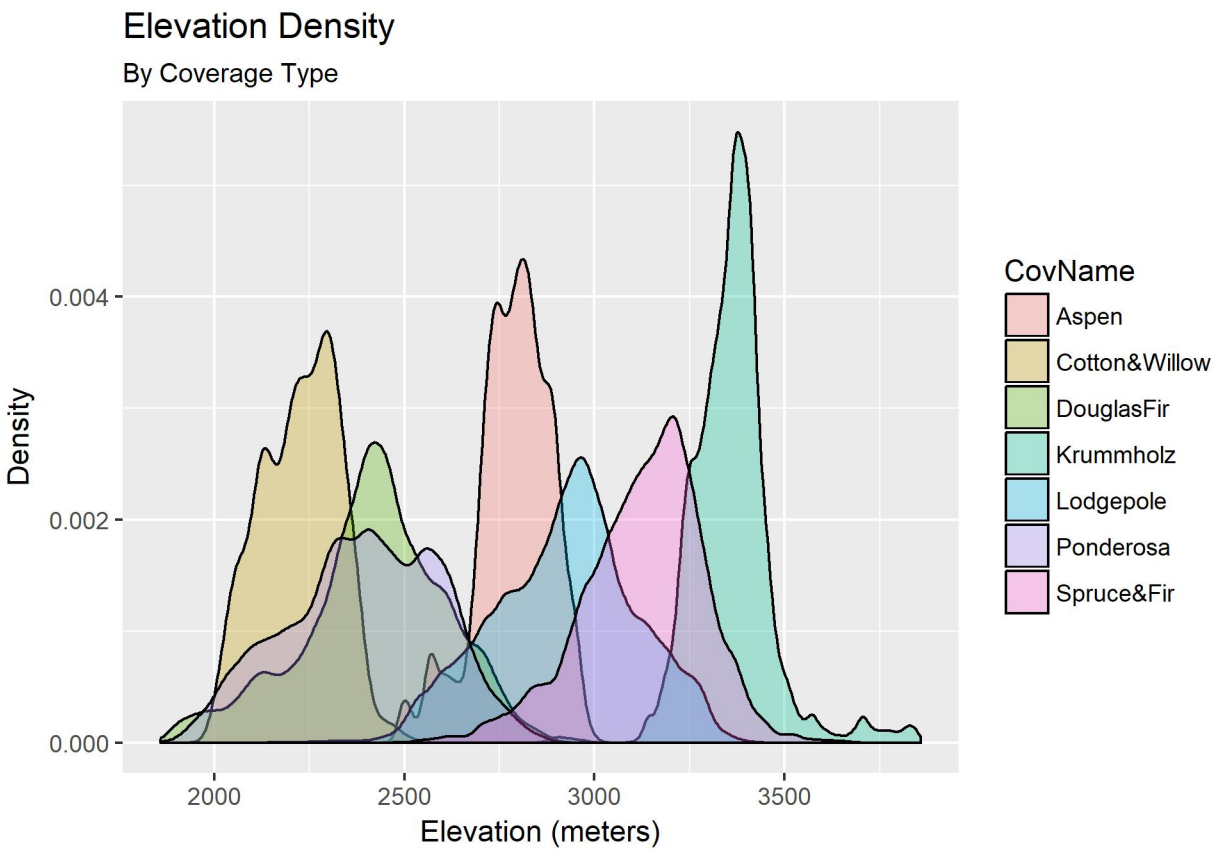

Figure 4: Elevation Density by Coverage Type

```
            x="Elevation (meters)", y="Density")
        # theme(axis.text.x = element_text(angle=-90))
ggsave("Fig-Elev-Density-CovName.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

The ranges of elevation for each coverage type is shown but with all the overlapping it is difficult to comprehend the different elevation ranges. Each coverage type will be graphed separately next to see if that makes the relative density ranges easier to see.

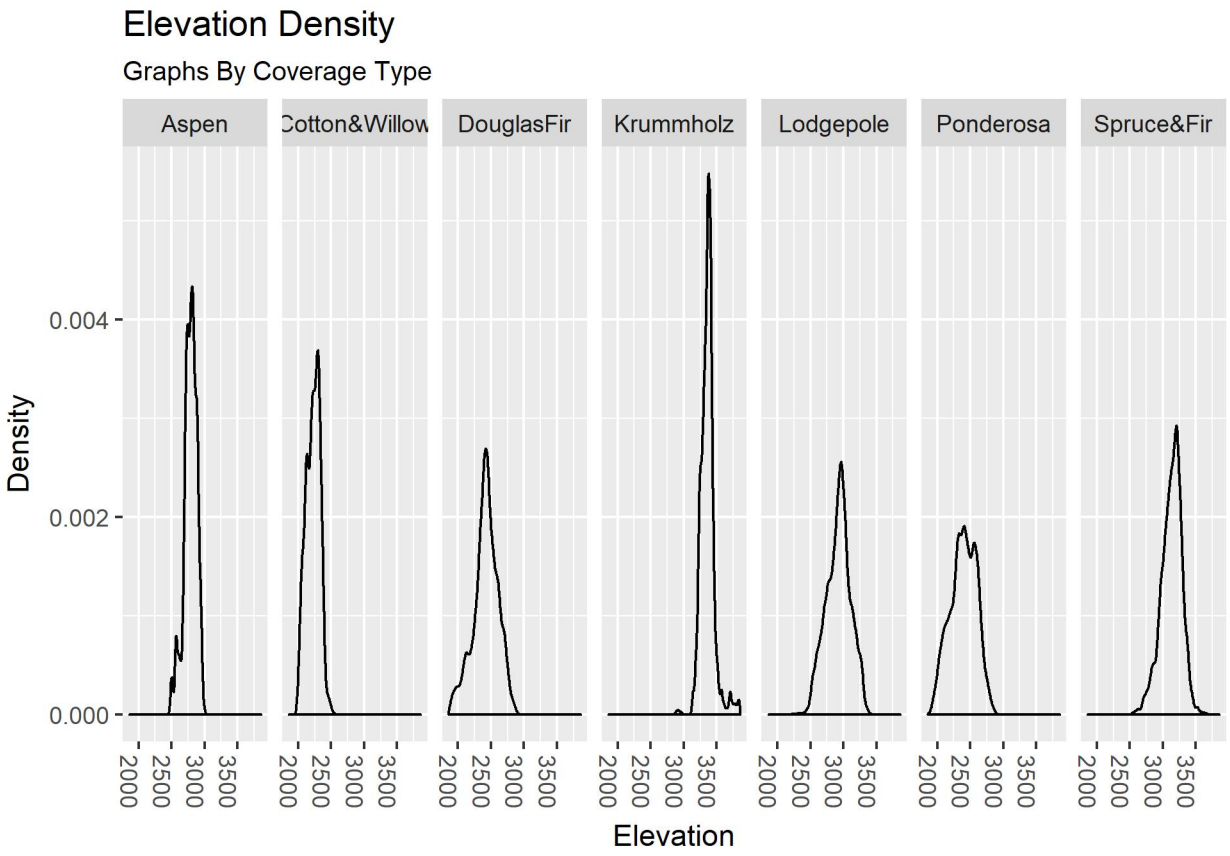

Figure 5: Elevation Density Coverage Type Graphs

Elevation Density Individual Graphs

```
g <- ggplot(forestcover,aes(Elev)) +
     geom_density()    +
     facet_grid(. ~ factor(CovName)) +
     labs(title = 'Elevation Density',
            subtitle = 'Graphs By Coverage Type',
            x="Elevation", y="Density") +
     theme(axis.text.x = element_text(angle=-90))
   # theme(axis.text.x = element_text(face="bold", color="#993333", size=14, angle=45))
ggsave("Fig-Elev-Density-CovType-Facets.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

This graph shows each coverage type clearly but it is still difficult to see how the ranges compare to each coverage type. The ggridges package provides another view.

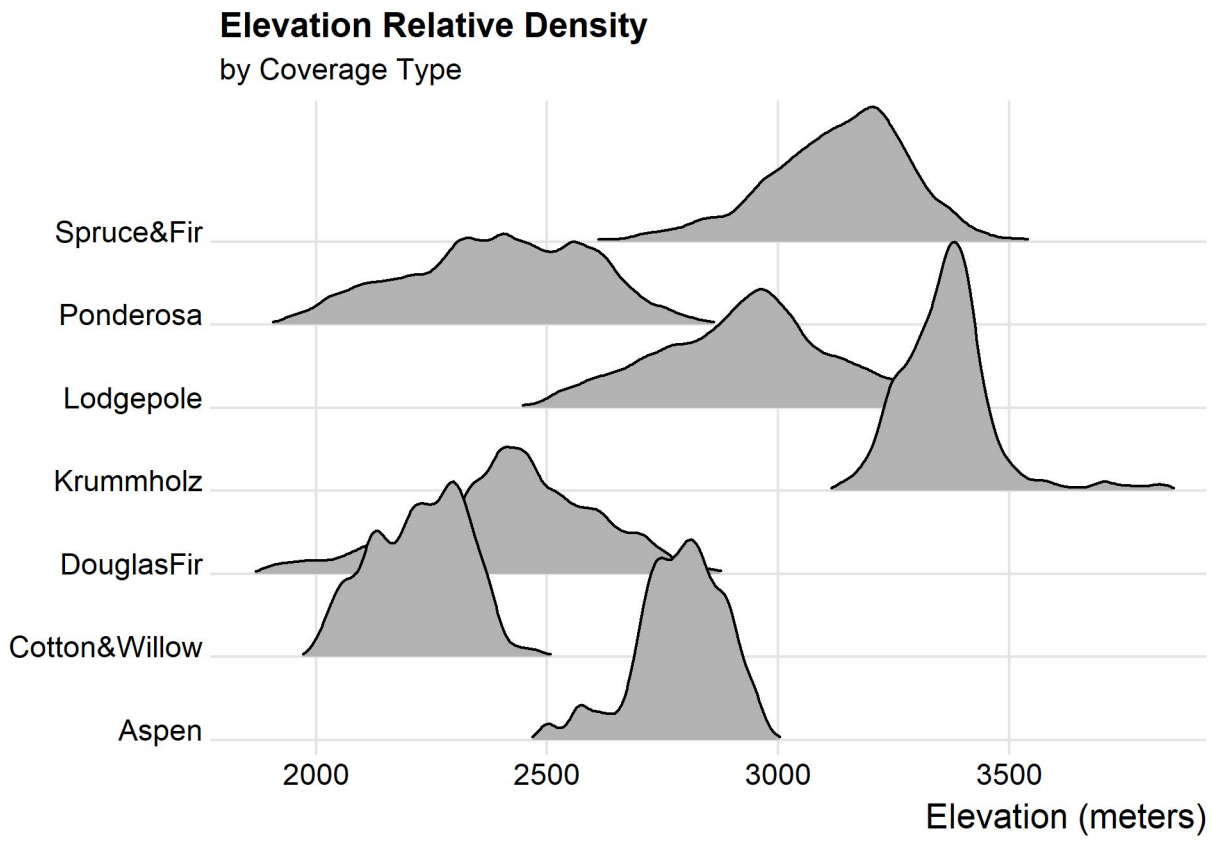

Figure 6: Elevation Density by Coverage Type - Offset

Elevation Density - Offset Graphs

```
g <- ggplot(forestcover,aes(Elev, CovName)) +
     geom_density_ridges(scale = 3, rel_min_height = 0.01) +
     scale_x_continuous(expand = c(0.01, 0)) +
     scale_y_discrete(expand = c(0.01, 0)) +
     labs(title = 'Elevation Relative Density',
     subtitle = 'by Coverage Type',
            x="Elevation (meters)") + # , y="Coverage Type") +
     theme_ridges(font_size = 13, grid = T) + theme(axis.title.y = element_blank())

ggsave("Fig-Elev-Density-CovType-Offset.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

```
## Picking joint bandwidth of 15.8
```

The density ridges geom gives the best feel for the ranges of elevation for each coverage type. It looks like the elevation is a significant factor in helping determine coverage type.

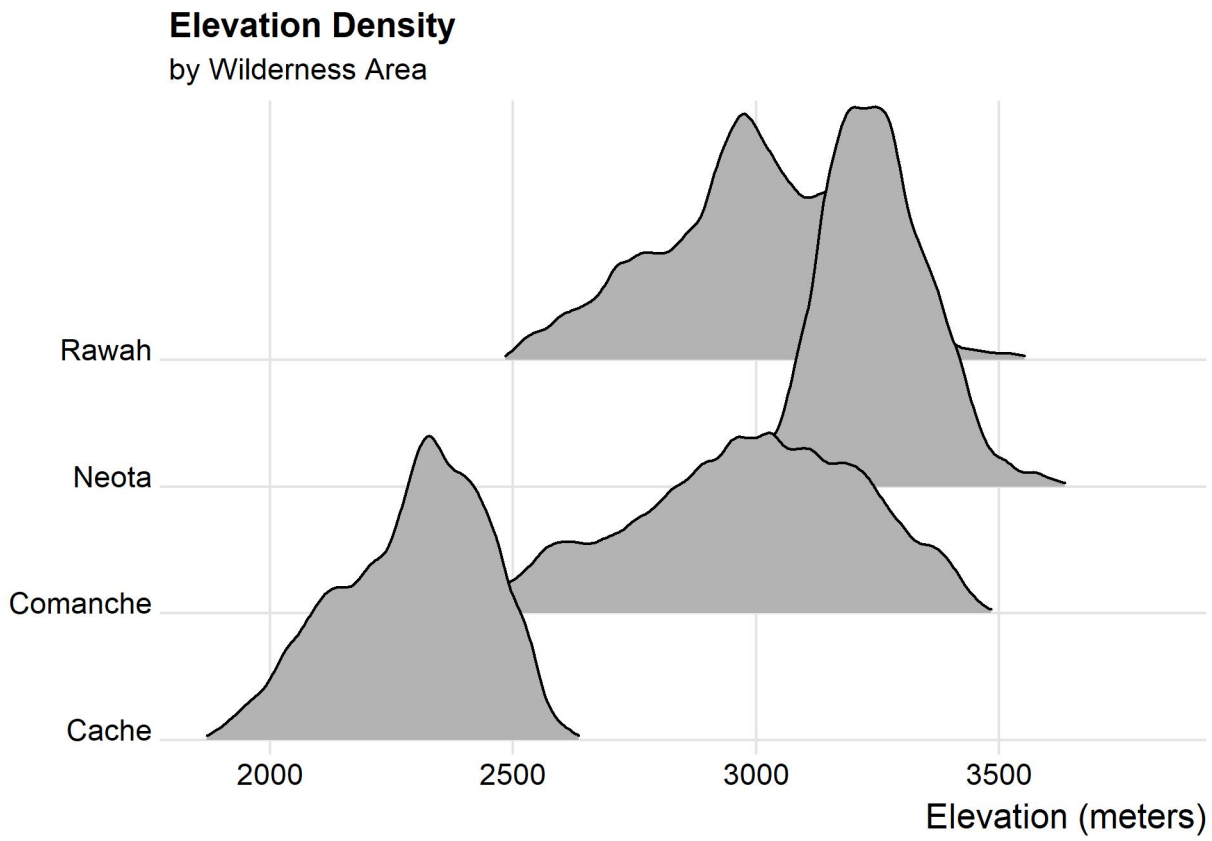

Figure 7: Elevation Density by Wilderness Area

Elevation Density by Wilderness Area

```
g <- ggplot(forestcover,aes(Elev, Wilderness_Area)) +
     geom_density_ridges(scale = 3, rel_min_height = 0.01) +
     scale_x_continuous(expand = c(0.01, 0)) +
     scale_y_discrete(expand = c(0.01, 0)) +
     labs(title = 'Elevation Density',
     subtitle = 'by Wilderness Area',
     x="Elevation (meters)") + # , y="Wilderness Area") +
     theme_ridges(font_size = 13, grid = T) + theme(axis.title.y = element_blank())
ggsave("Fig-Elev-Density-WildArea.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

```
## Picking joint bandwidth of 15.6
```

This shows that the elevation varies by wilderness area as well, which could be expected since each wilderness area will have it's unique topography.

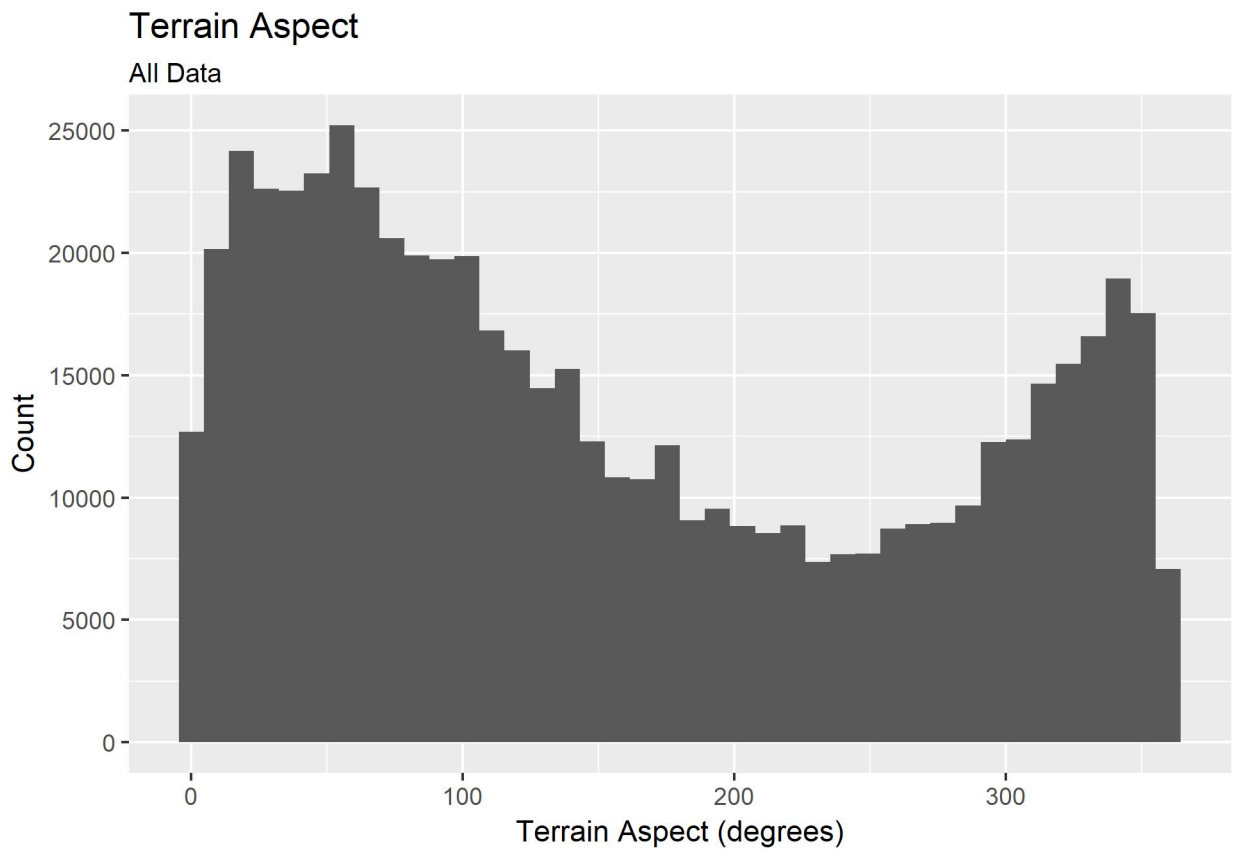

Figure 8: Terrain Aspect Histogram

Terrain Aspect Histogram

```
g <- ggplot(forestcover,aes(Aspect)) +
     geom_histogram(bins=40) +
     labs(title = 'Terrain Aspect',
            subtitle = 'All Data',
            x="Terrain Aspect (degrees)", y="Count")
       # theme(axis.text.x = element_text(angle=-90))
ggsave("Fig-Aspect-Histogram.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

The aspect, the direction the terrain is facing, forms an interesting sine wave with the most terrain facing 45 degrees or North East and the least amount of terrain facing 230 degrees or South West. # There are spikes in the graph that look like anomalies due to the size of the bin used by the #histogram. There are dips in the bins next to the spikes.

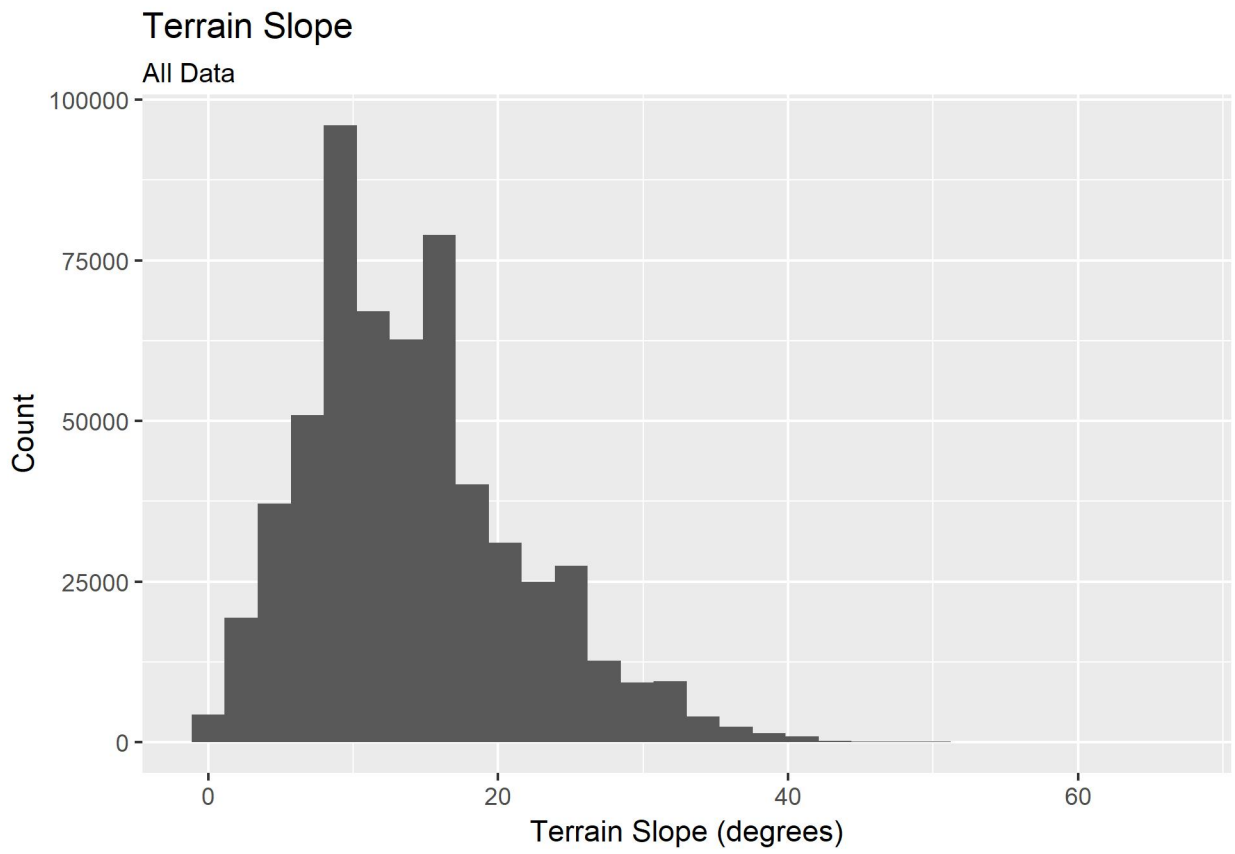

Figure 9: Terrain Slope Histogram

Terrain Slope Histogram

```
g <- ggplot(forestcover,aes(Slope)) +
     geom_histogram() +
     labs(title = 'Terrain Slope',
            subtitle = 'All Data',
            x="Terrain Slope (degrees)", y="Count")
       # theme(axis.text.x = element_text(angle=-90))
ggsave("Fig-Slope-Histogram.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

The slope distribution seems reasonable and smooth with most of the trees on terrain sloping from 3 degrees to 30 degrees and very few trees on slopes greater than 40 degrees.

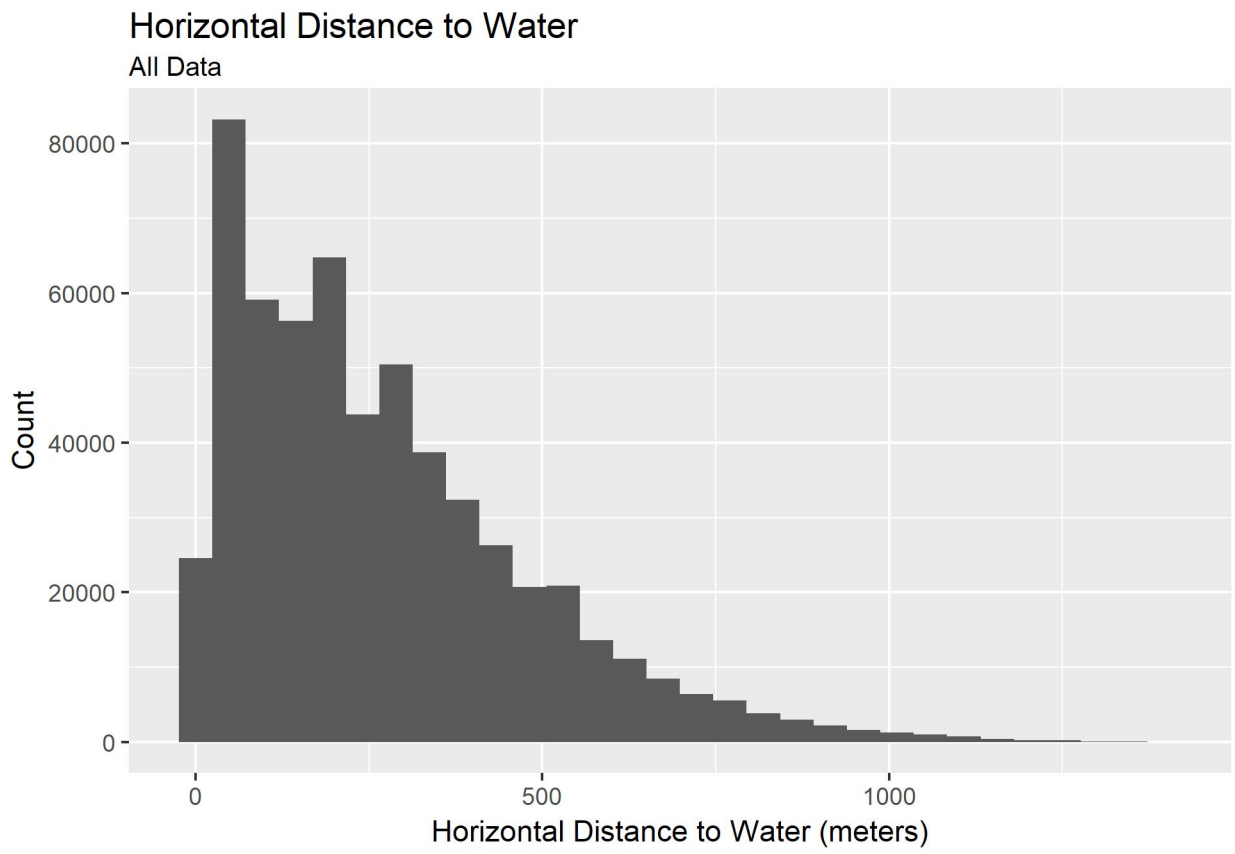

Figure 10: Horizontal Distance to Water Histogram

Horizontal Distance to Water Histogram

```
g <- ggplot(forestcover,aes(H2OHD)) +
     geom_histogram() +
     labs(title = 'Horizontal Distance to Water',
            subtitle = 'All Data',
            x="Horizontal Distance to Water (meters)", y="Count")
       # theme(axis.text.x = element_text(angle=-90))
ggsave("Fig-Horizontal-Water-Dist-Histogram.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

The horizontal distance to water features seems reasonable. Most of the trees are fairly close to water which shows trees like water.

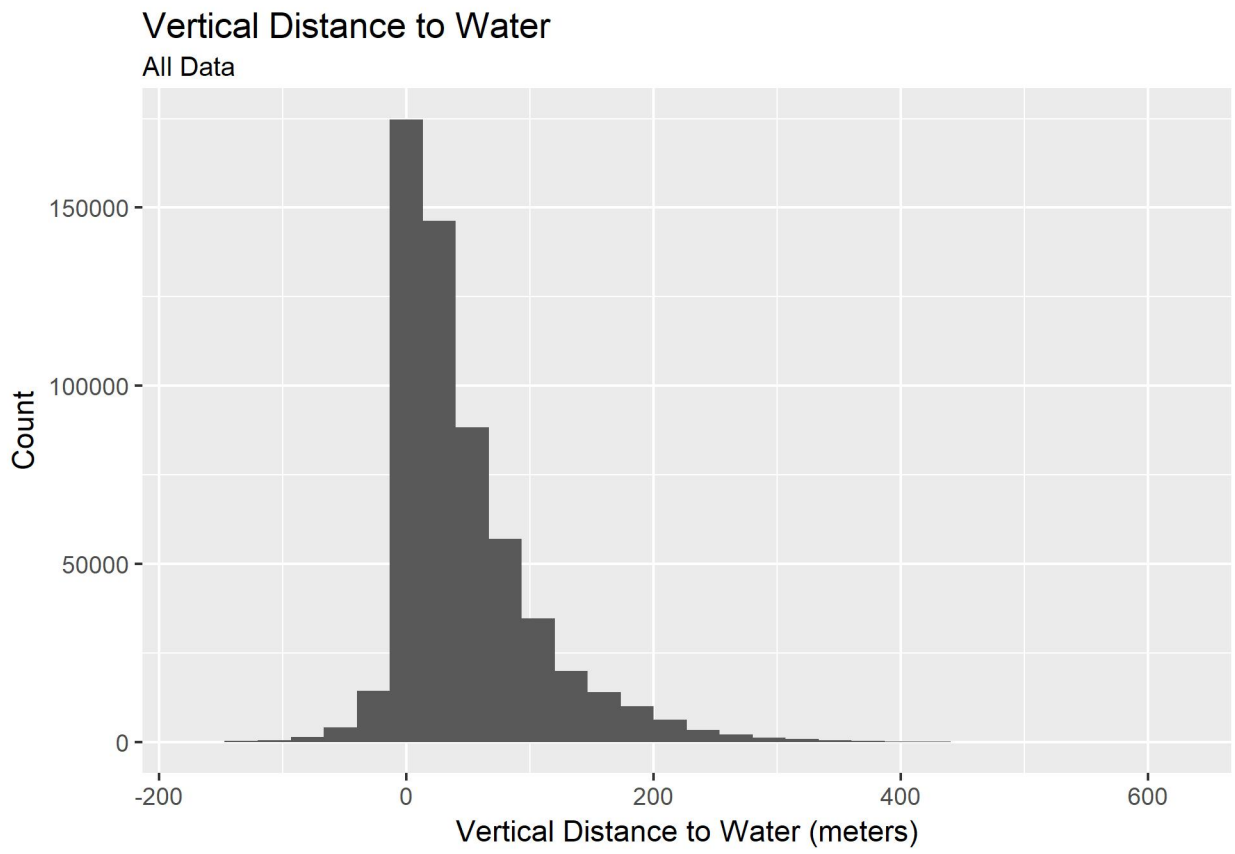

Figure 11: Vertical Distance to Water Histogram

Vertical Distance to Water Histogram

```
g <- ggplot(forestcover,aes(H2OVD)) +
     geom_histogram() +
     labs(title = 'Vertical Distance to Water',
            subtitle = 'All Data',
            x="Vertical Distance to Water (meters)", y="Count")
       # theme(axis.text.x = element_text(angle=-90))
ggsave("Fig-Vertical-Water-Dist-Histogram.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

The vertical distance to the nearest water also seems reasonable with the majority of trees having water from above within 5 to 150 meters. For some trees the nearest water is below within about 40 meters.

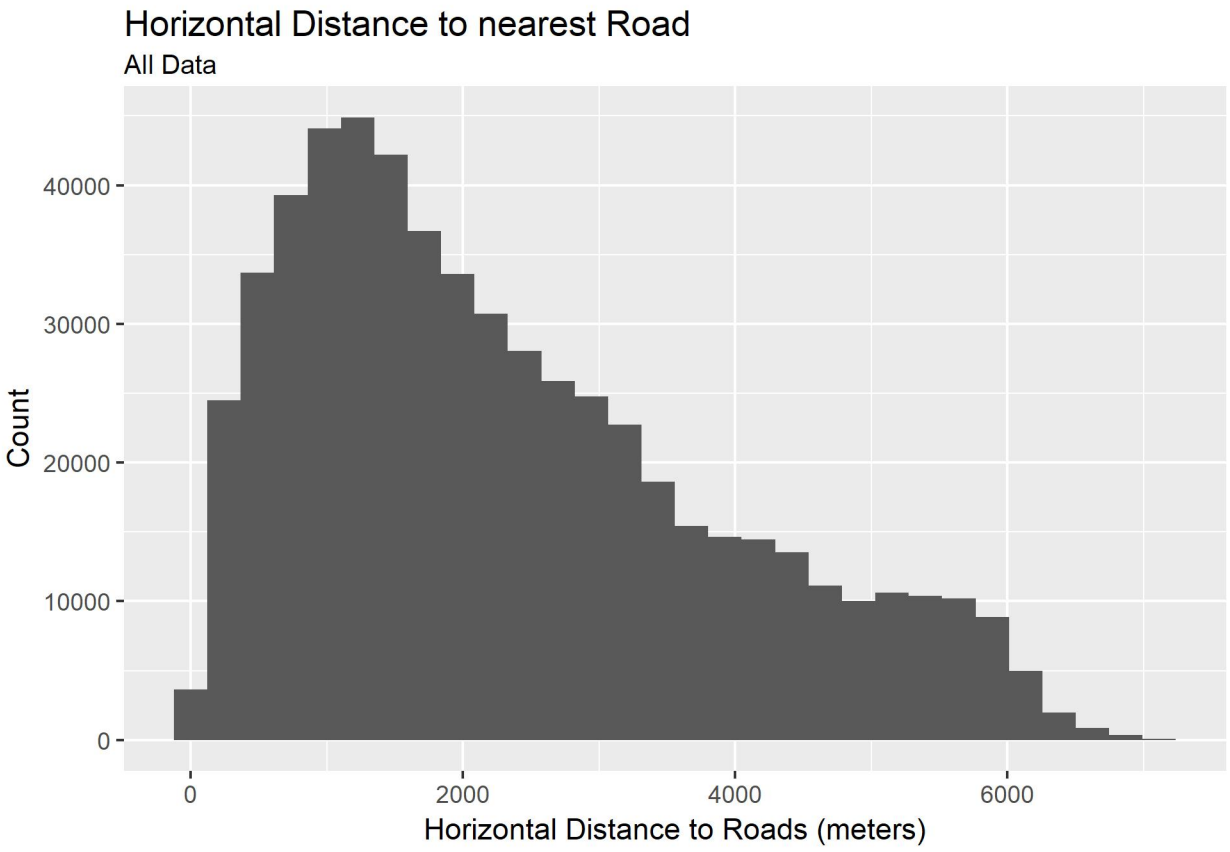

Figure 12: Horizontal Distance to Roads Histogram

Horzontal Distance to Roads

```
g <- ggplot(forestcover,aes(RoadHD)) +
     geom_histogram() +
     labs(title = 'Horizontal Distance to nearest Road',
            subtitle = 'All Data',
            x="Horizontal Distance to Roads (meters)", y="Count")
       # theme(axis.text.x = element_text(angle=-90))
ggsave("Fig-Horizontal-Road-Dist-Histogram.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

The nearest roads are closer than I would have expected for wilderness areas. Most of the trees are within 1700 meters or about a mile of a road.

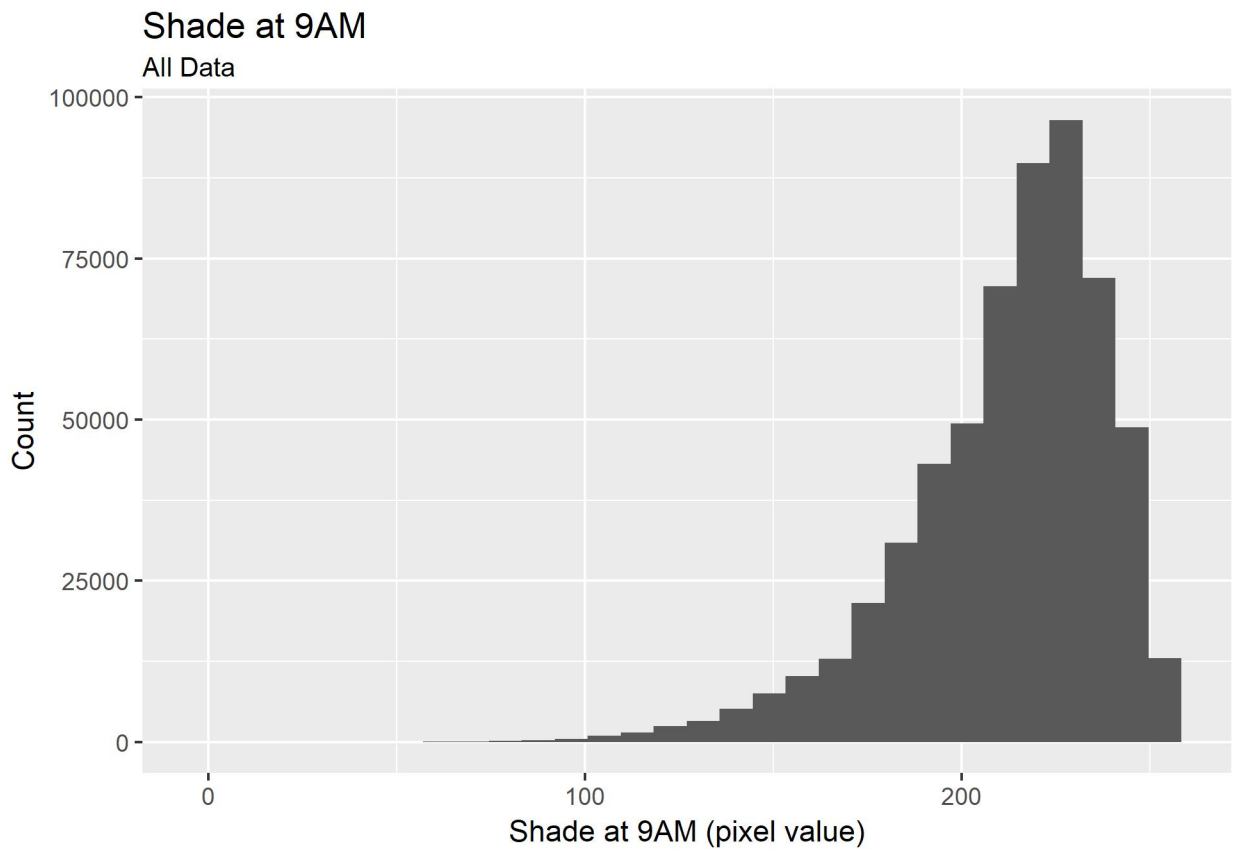

Figure 13: Shade at 9AM Histogram

Shade at 9AM Histogram

```
g <- ggplot(forestcover,aes(Shade9AM)) +
     geom_histogram() +
     labs(title = 'Shade at 9AM',
            subtitle = 'All Data',
            x="Shade at 9AM (pixel value)", y="Count")
       # theme(axis.text.x = element_text(angle=-90))
ggsave("Fig-Shade9AM-Histogram.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

The shade value is fairly high at 9am with most area having between 80% shade (207/254 pixel value) and 90% shade (238/254 pixel value).

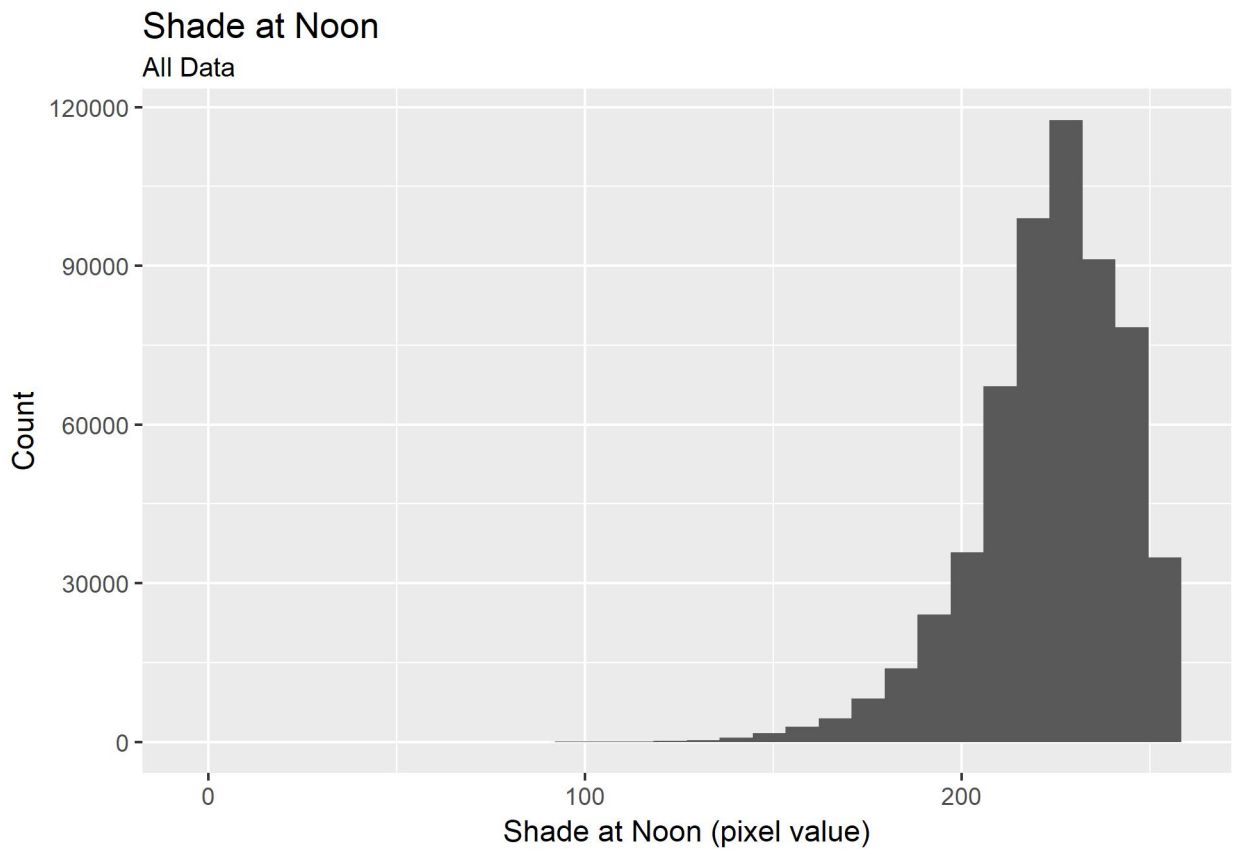

Figure 14: Shade at Noon Histogram

Shade at 12PM Noon Histogram

```
g <- ggplot(forestcover,aes(Shade12PM)) +
     geom_histogram() +
     labs(title = 'Shade at Noon',
            subtitle = 'All Data',
            x="Shade at Noon (pixel value)", y="Count")
       # theme(axis.text.x = element_text(angle=-90))
ggsave("Fig-ShadeNoon-Histogram.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

There is even more shade at noon with most areas having between 82% (210/254) and 100% shade (252/254).

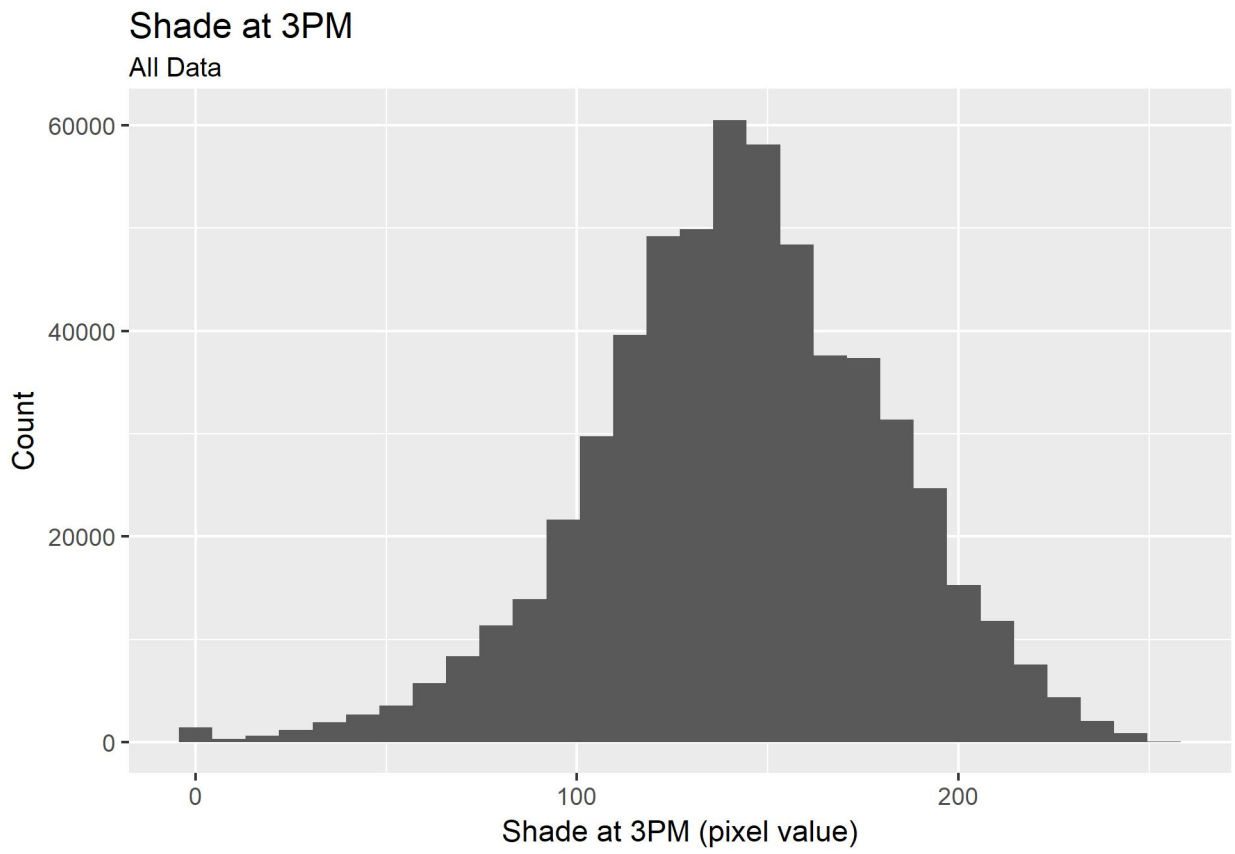

Figure 15: Shade at 3PM Histogram

Shade at 3PM Histogram

```
g <- ggplot(forestcover,aes(Shade3PM)) +
     geom_histogram() +
     labs(title = 'Shade at 3PM',
            subtitle = 'All Data',
            x="Shade at 3PM (pixel value)", y="Count")
       # theme(axis.text.x = element_text(angle=-90))
ggsave("Fig-Shade3PM-Histogram.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

There is more sun (less shade) at 3pm with most cells having between 37% (94/254) and 77% (196/254) shade.

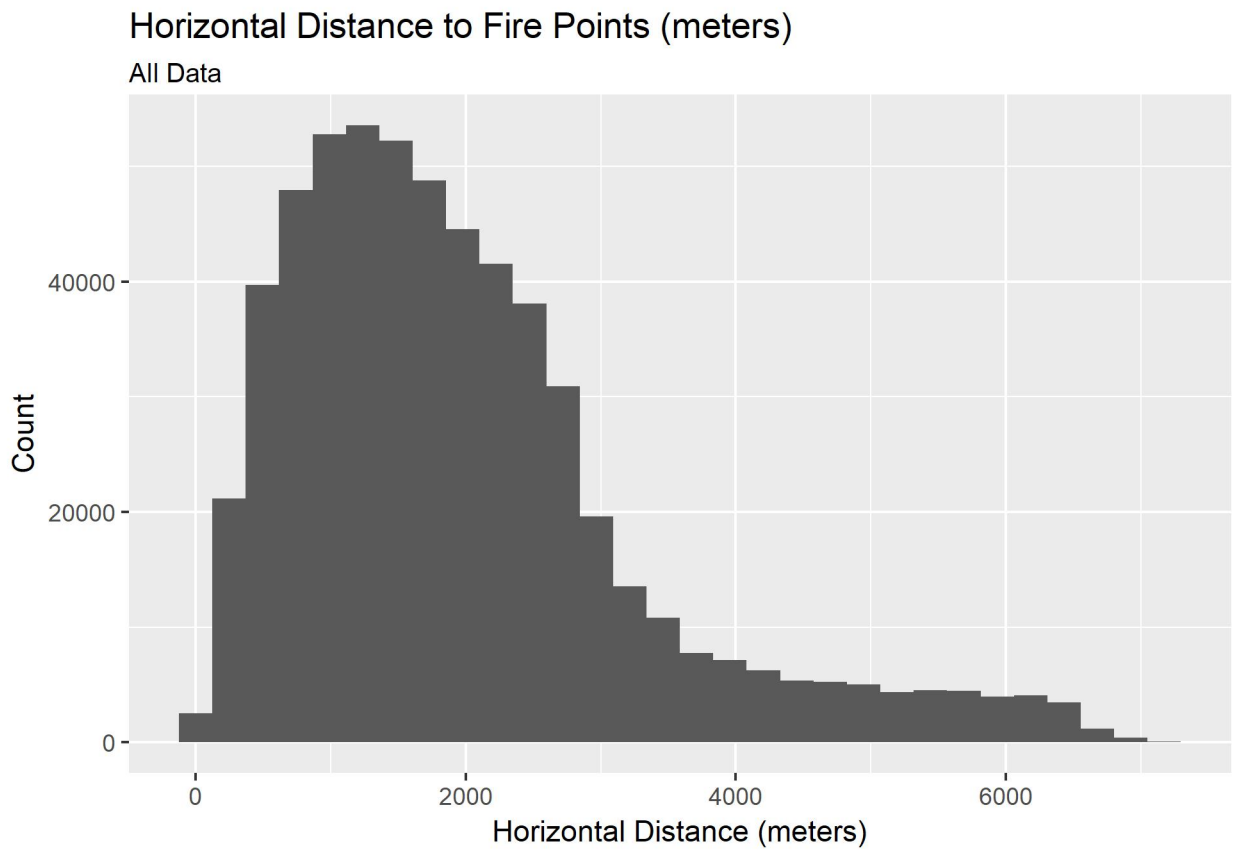

Figure 16: Horizontal Distance to Fire Points Histogram

Horzontal Distance to Fire Points

```
g <- ggplot(forestcover,aes(FirePtHD)) +
     geom_histogram() +
     labs(title = 'Horizontal Distance to Fire Points (meters)',
            subtitle = 'All Data',
            x="Horizontal Distance (meters)", y="Count")

ggsave("Fig-FirePtHD-Histogram.jpg")
```

```
## Saving 6.5 x 4.5 in image
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

The distance from fire occurrence is similar to the distance to roads graph.

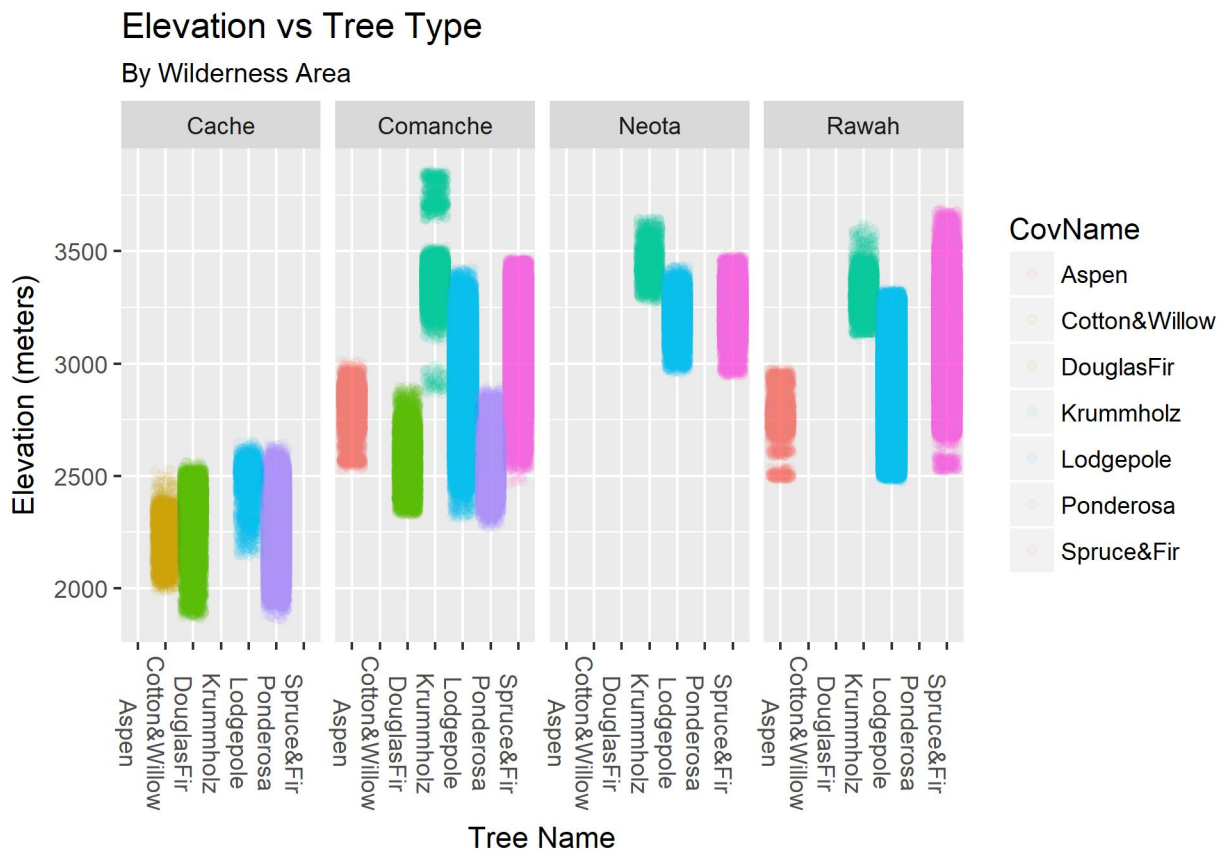

Figure 17: Elevations vs Tree Type and Wilderness Area

Examining Relationships between Data

Now that the histograms of continuous data has been shown and intuitively looks reasonable, various dataare compared to see how they might help predict the coverage type.

Elevations vs Tree Type and Wilderness Area

```
g <- ggplot(forestcover,aes(CovName,Elev,col=CovName)) +
      geom_jitter(alpha=alphaVal) +
      facet_grid(. ~ Wilderness_Area) +
      labs(title = 'Elevation vs Tree Type',
            subtitle = 'By Wilderness Area',
            x="Tree Name", y="Elevation (meters)")  +
      theme(axis.text.x = element_text(angle=-90))
ggsave("Fig-Elev-TreeType-WildArea.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

The interesting elevation gap in Krummholz tree occurs in the Comanche wilderness area. But we see there are two gaps and must be due to some areas of terrain that vary significantly in the Comanche wilderness area. I was expecting that the Krummholz tree type should have been continuous in each wilderness area and the gap would be explained by different elevation ranges in different wilderness areas. This plot also shows that there are 1 to 4 types of trees missing in each wilderness area.

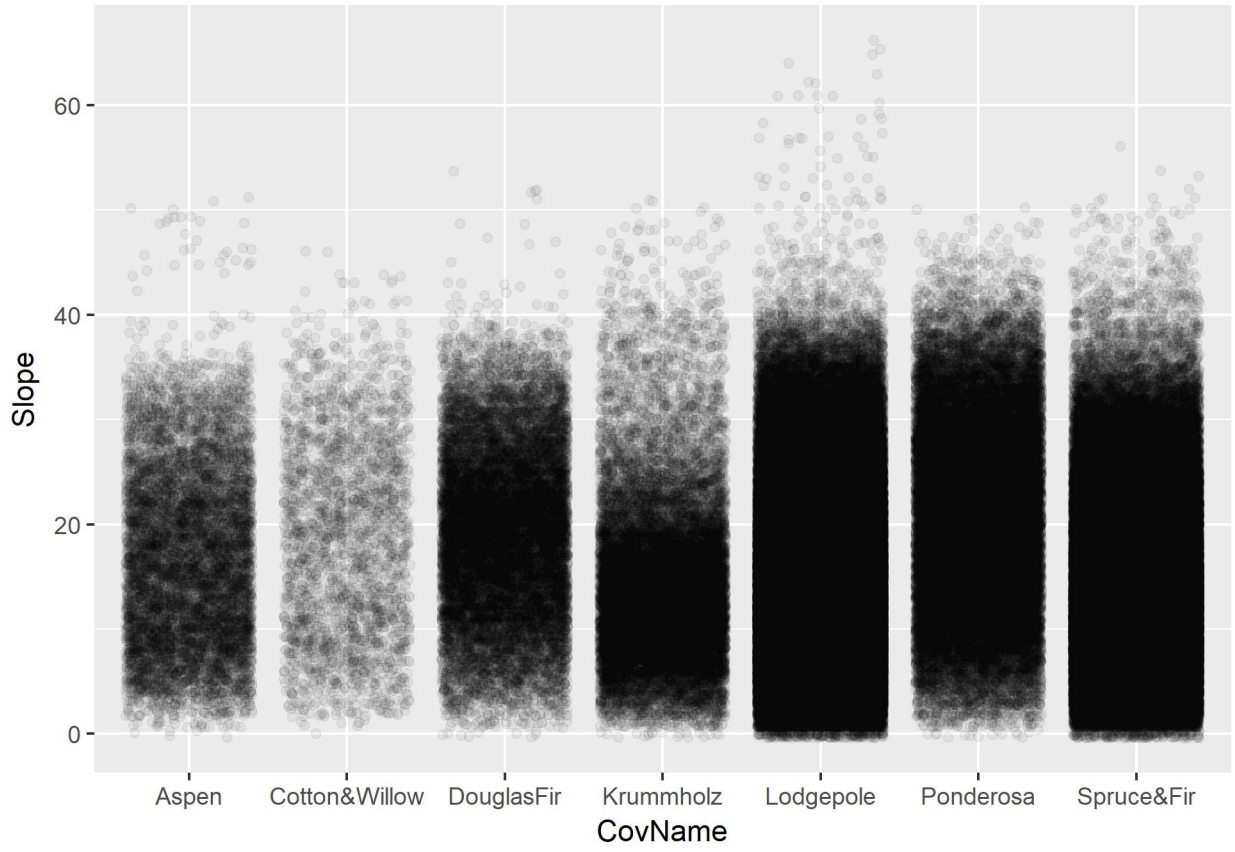

Figure 18: Slope vs Tree Type

Slope vs Tree Type

```
g <- ggplot(forestcover,aes(CovName,Slope)) +
        geom_jitter(alpha=alphaVal)
  ggsave("Fig-Slope-Vs-TreeType.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

The distribution of tree type by slope appears to be pretty evenly distributed and the distribution is close to the same for each tree type. It doesn't look like slope will be a major factor to help determine the tree type.

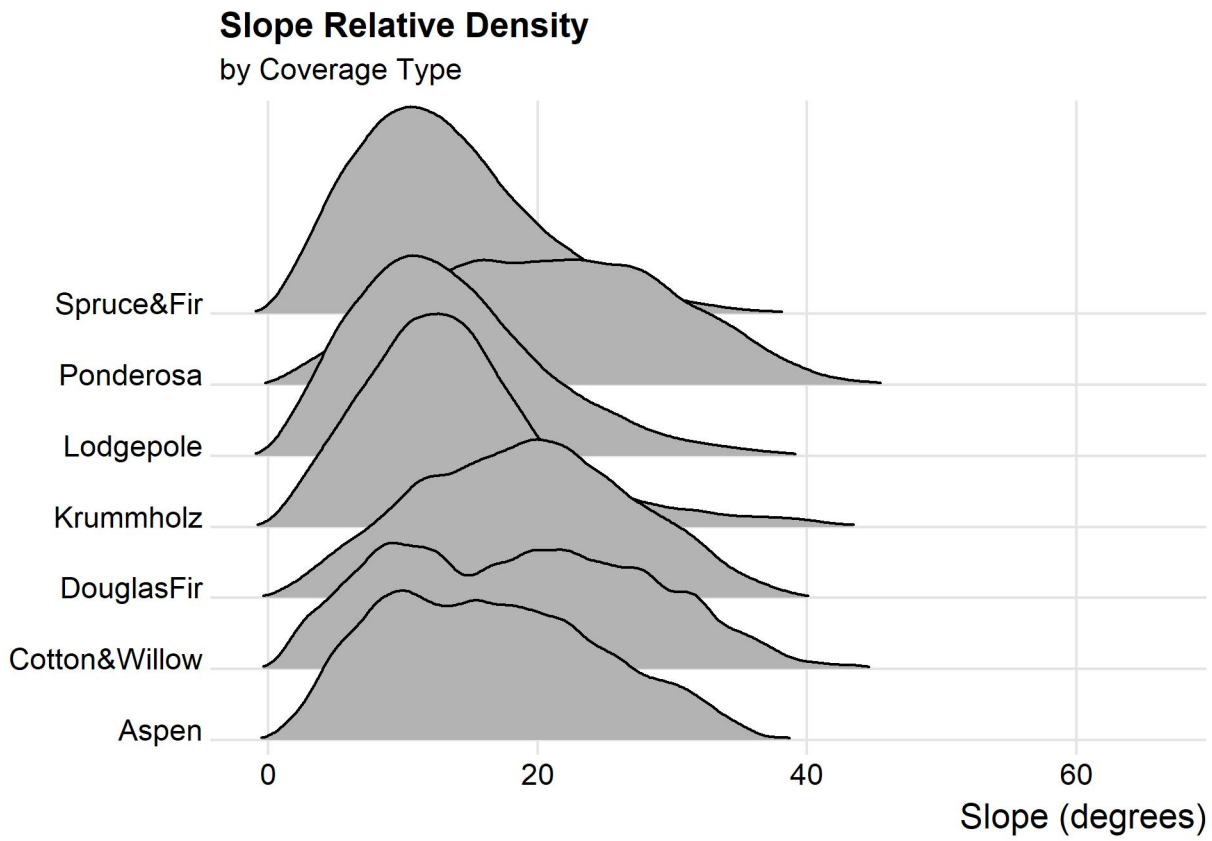

Figure 19: Slope Density by Coverage Type - Offset

Slope vs Tree Type using Ridges

```
g <- ggplot(forestcover,aes(Slope, CovName)) +
     geom_density_ridges(scale = 3, rel_min_height = 0.01) +
     scale_x_continuous(expand = c(0.01, 0)) +
     scale_y_discrete(expand = c(0.01, 0)) +
     labs(title = 'Slope Relative Density',
          subtitle = 'by Coverage Type',
          x="Slope (degrees)"
          # , y="Coverage Type"
     ) +
     theme_ridges(font_size = 13, grid = T) + theme(axis.title.y = element_blank())

ggsave("Fig-Slope-Density-CovType-Offset.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

```
## Picking joint bandwidth of 0.97
```

The ggridges geom gives a better view of slope distribution by coverage type than the jitter geom. It is much clearer here that the slope distributions is similar for each tree type and that it will not be that helpful in determining coverage type.

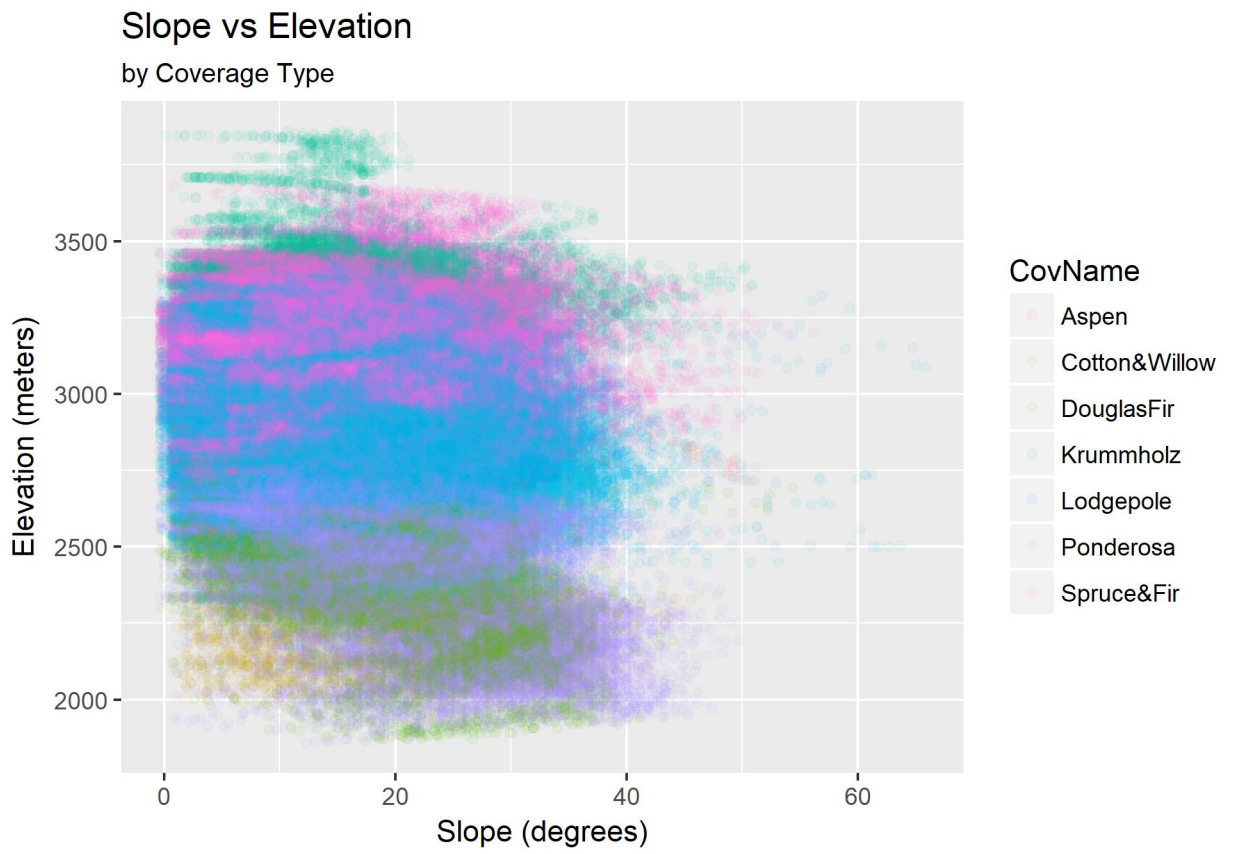

Figure 20: Slope vs Elev with Tree Type

Slope vs Elev with Tree Type

```
g <- ggplot(forestcover,aes(Slope,Elev,col=CovName)) +
    geom_jitter(alpha=alphaVal) +
    labs(title = 'Slope vs Elevation',
          subtitle = 'by Coverage Type',
          x="Slope (degrees)"
          , y="Elevation (meters)"
     )
ggsave("Fig-Slope-Vs-Elev-By-TreeType-jitter.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

There is a scattering of trees continuously over the range of the slope and elevation. It is hard to interpret this graph and draw any conclusions.

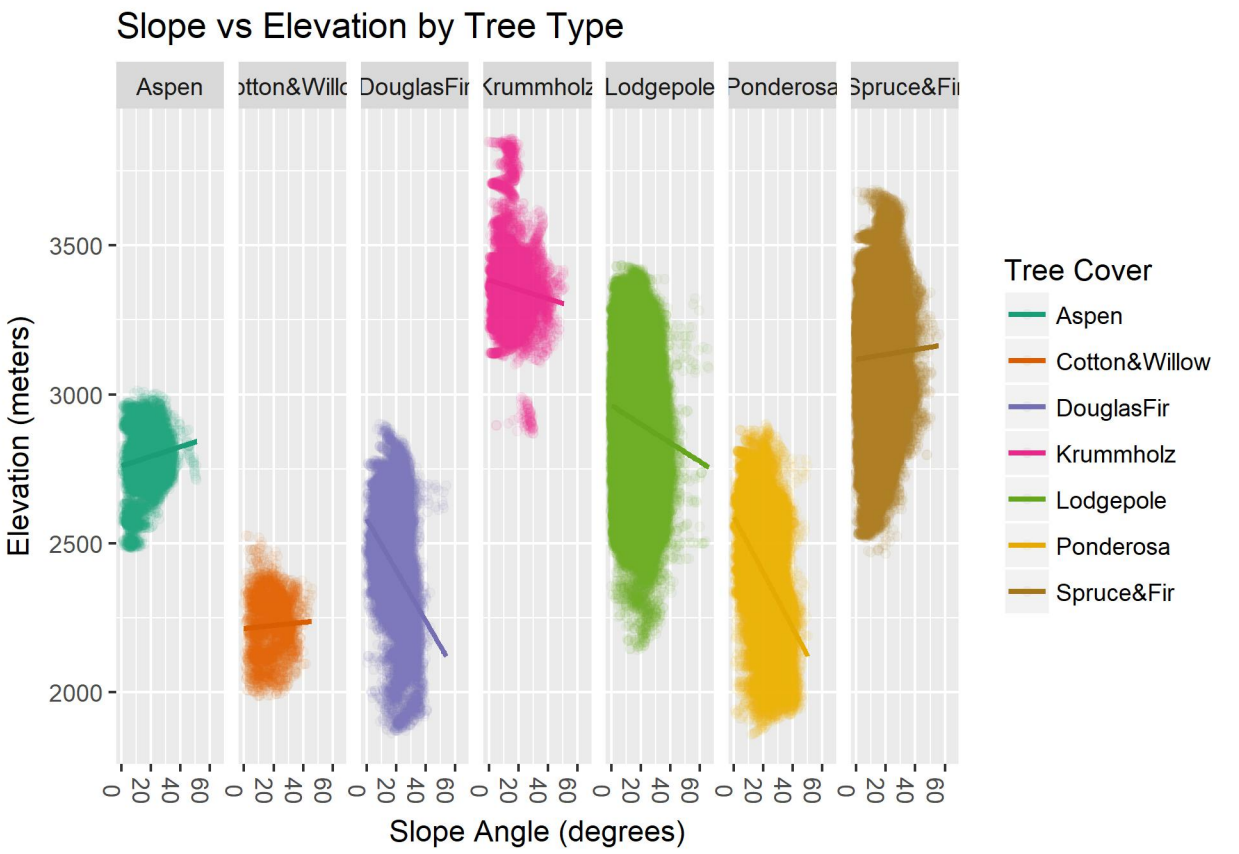

Figure 21: Elevation vs Slope and Tree Type

Elevation vs Slope by Tree Type

```
library(RColorBrewer)
myColors1 <- c(brewer.pal(7, "Dark2"), "black")

g <- ggplot(forestcover,aes(Slope,Elev,col=CovName)) +
     geom_jitter(alpha=alphaVal) +
     stat_smooth(method = "lm", se = F) +
     facet_grid(. ~ CovName) +
     scale_color_manual("Tree Cover",values=myColors1) +
     labs(title="Slope vs Elevation by Tree Type",
          x = "Slope Angle (degrees)",
          y = "Elevation (meters)") +
     theme(axis.text.x = element_text(angle=-90))
ggsave("Fig-Slope-Vs-Elev-by-CovName-facets.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

This graph replicates graph 12, so, no new info here. Adding the smooth line mainly helps to see the color of the legend when the alpha value is so low.

Elevation vs Terrain Aspect

```
myColors1 <- c(brewer.pal(7, "Dark2"), "black")
  g <- ggplot(forestcover,aes(Aspect,Elev,col=CovName)) +
       geom_jitter(alpha=alphaVal) +
       stat_smooth(method = "lm", se = F) +
#       facet_grid(. ~ CovName) +
       scale_color_manual("Tree Cover",values=myColors1) +
       labs(title="Elevation vs Terrain Aspect",
            x = "Aspect Direction (degrees)",
            y = "Elevation (meters)")
  ggsave("Fig-Elev-vs-Aspect-jitter.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

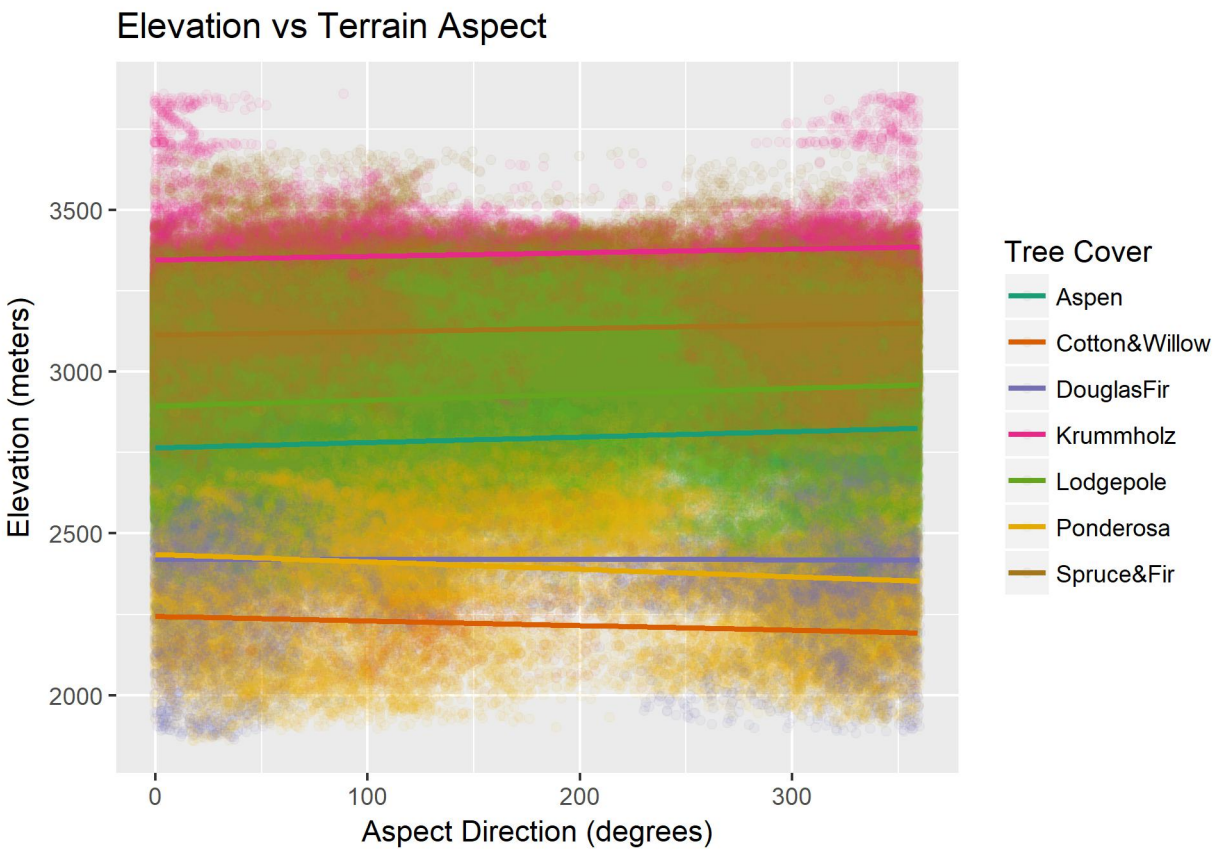

Figure 22: Elevation vs Terrain Aspect

The elevation vs aspect shows that, like slope, tree type is not related much to Aspect.

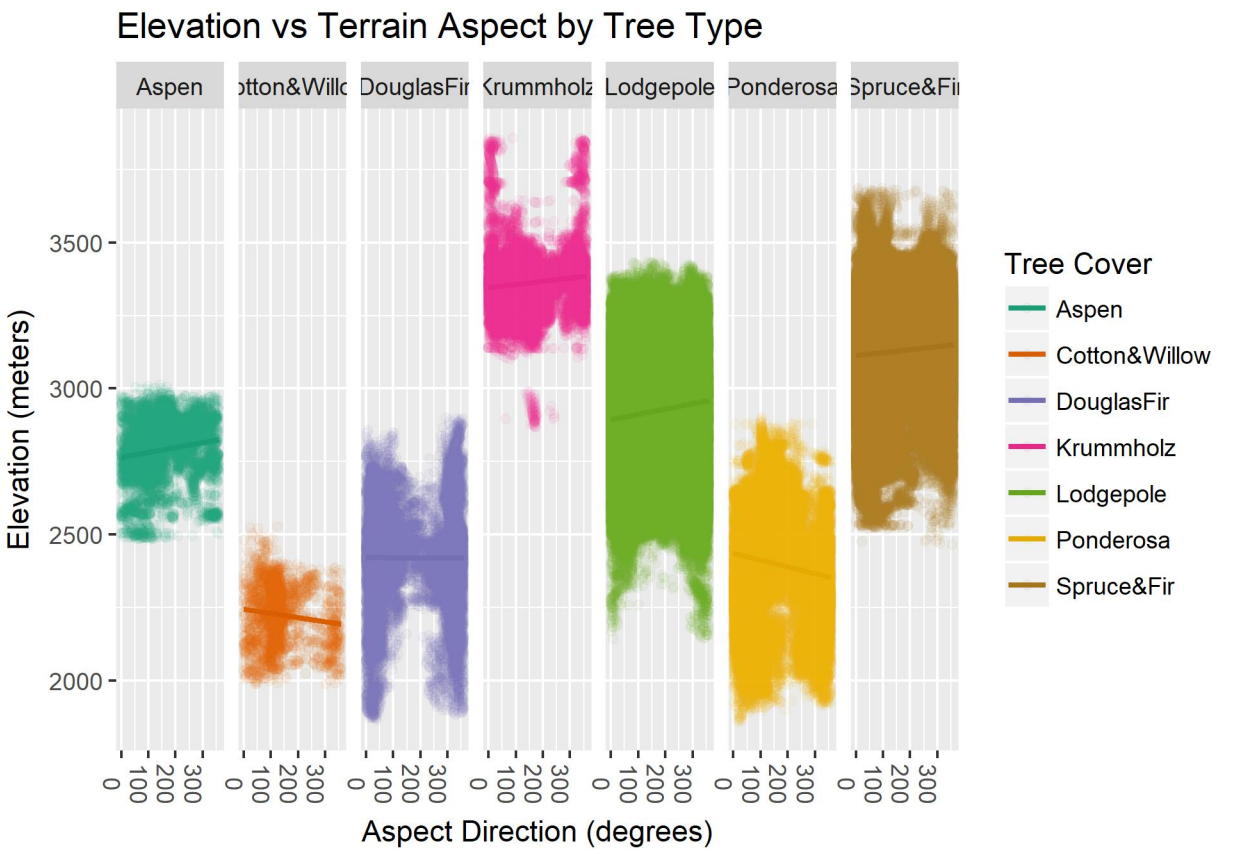

Figure 23: Elevation vs Terrain Aspect by Tree Type

Elevation vs Terrain Aspect by Tree Type - Figure 18

```
g <- ggplot(forestcover,aes(Aspect,Elev,col=CovName)) +
     geom_jitter(alpha=alphaVal) +
     stat_smooth(method = "lm", se = F) +
     facet_grid(. ~ CovName) +
     scale_color_manual("Tree Cover",values=myColors1) +
     labs(title="Elevation vs Terrain Aspect by Tree Type",
          x = "Aspect Direction (degrees)",
         y = "Elevation (meters)") +
     theme(axis.text.x = element_text(angle=-90))
ggsave("Fig-Elev-Vs-TerrainAspect-by-Treeytype-jitter.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

Another view of elevation vs aspect shows there are concentrations of tree types near 0 and 360 degrees. This occurs for all tree types and shows again that aspect is not going to help very much.

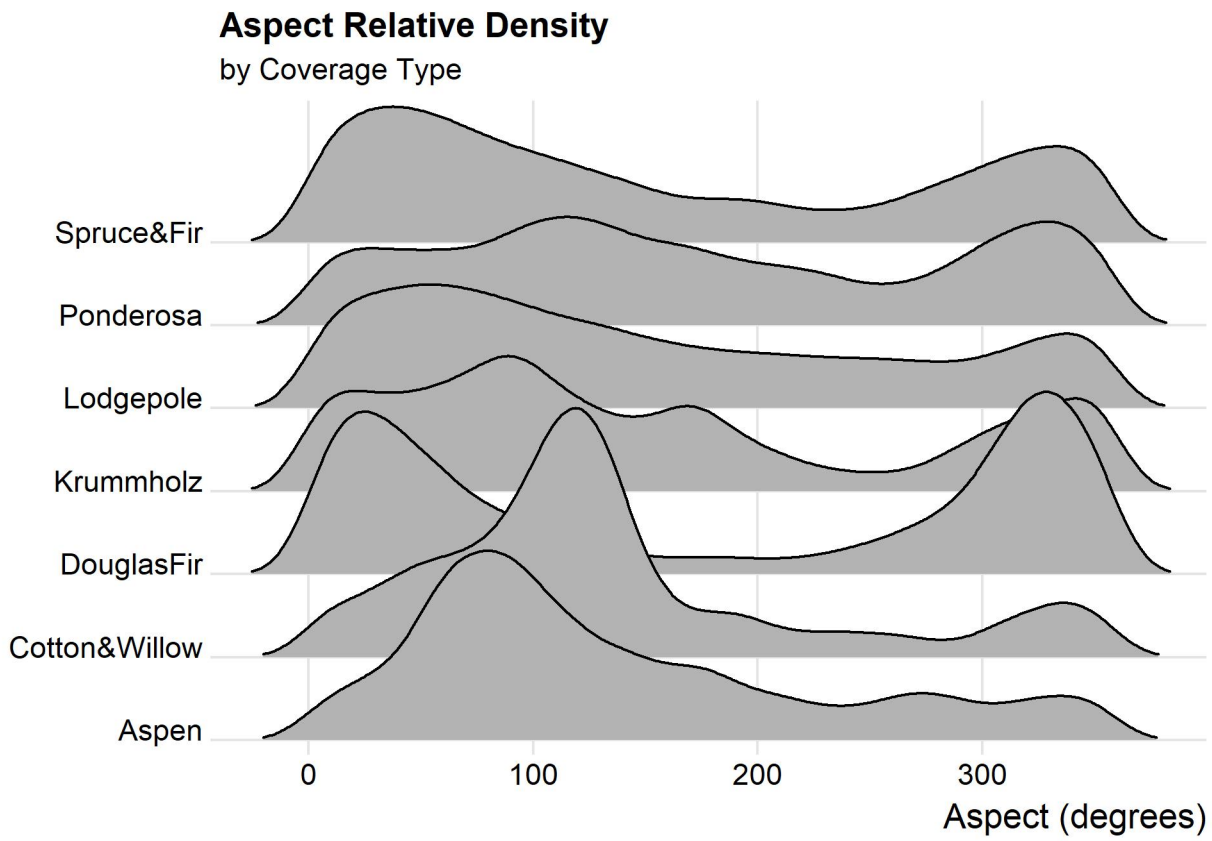

Figure 24: Aspect Density by Coverage Type - Offset

```
g <- ggplot(forestcover,aes(Aspect, CovName)) +
     geom_density_ridges(scale = 3, rel_min_height = 0.01) +
     scale_x_continuous(expand = c(0.01, 0)) +
     scale_y_discrete(expand = c(0.01, 0)) +
     labs(title = 'Aspect Relative Density',
          subtitle = 'by Coverage Type',
          x="Aspect (degrees)"
          # , y="Coverage Type"
     ) +
     theme_ridges(font_size = 13, grid = T) + theme(axis.title.y = element_blank())

ggsave("Fig-Aspect-Density-CovType-Offset.jpg")
```

```
## Saving 6.5 x 4.5 in image
## Picking joint bandwidth of 11.8
```

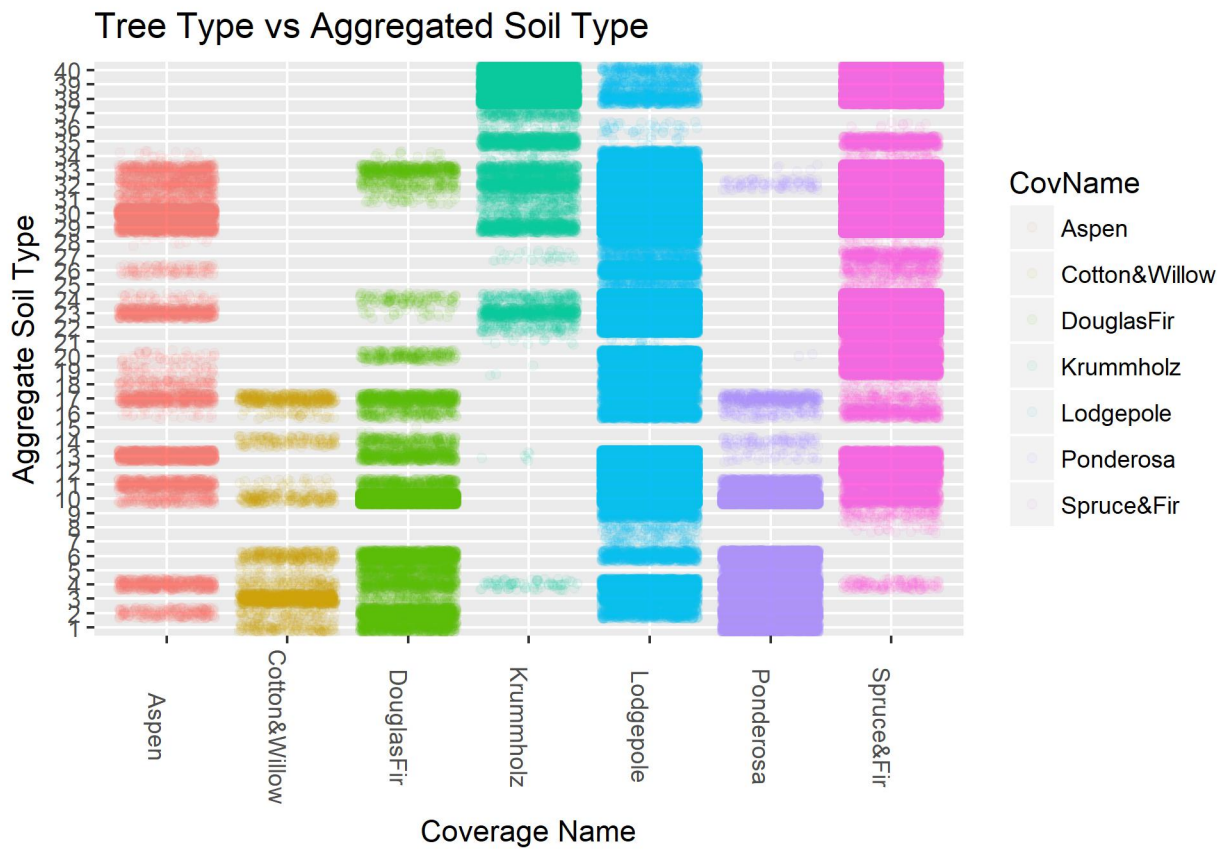

Figure 25: Tree Type vs Aggregated Soil Type

Tree Type vs Aggregated Soil Type

```
g <- ggplot(forestcover,aes(CovName,factor(SoilType), col=CovName)) +
      geom_jitter(alpha=alphaVal) +
      labs(title="Tree Type vs Aggregated Soil Type",
           x = "Coverage Name",
           y = "Aggregate Soil Type") +
      theme(axis.text.x = element_text(angle=-90))
ggsave("Fig-Tree-SoilType-jitter.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

Looking at aggregated soil types 1 through 40. Still nothing jumping out.

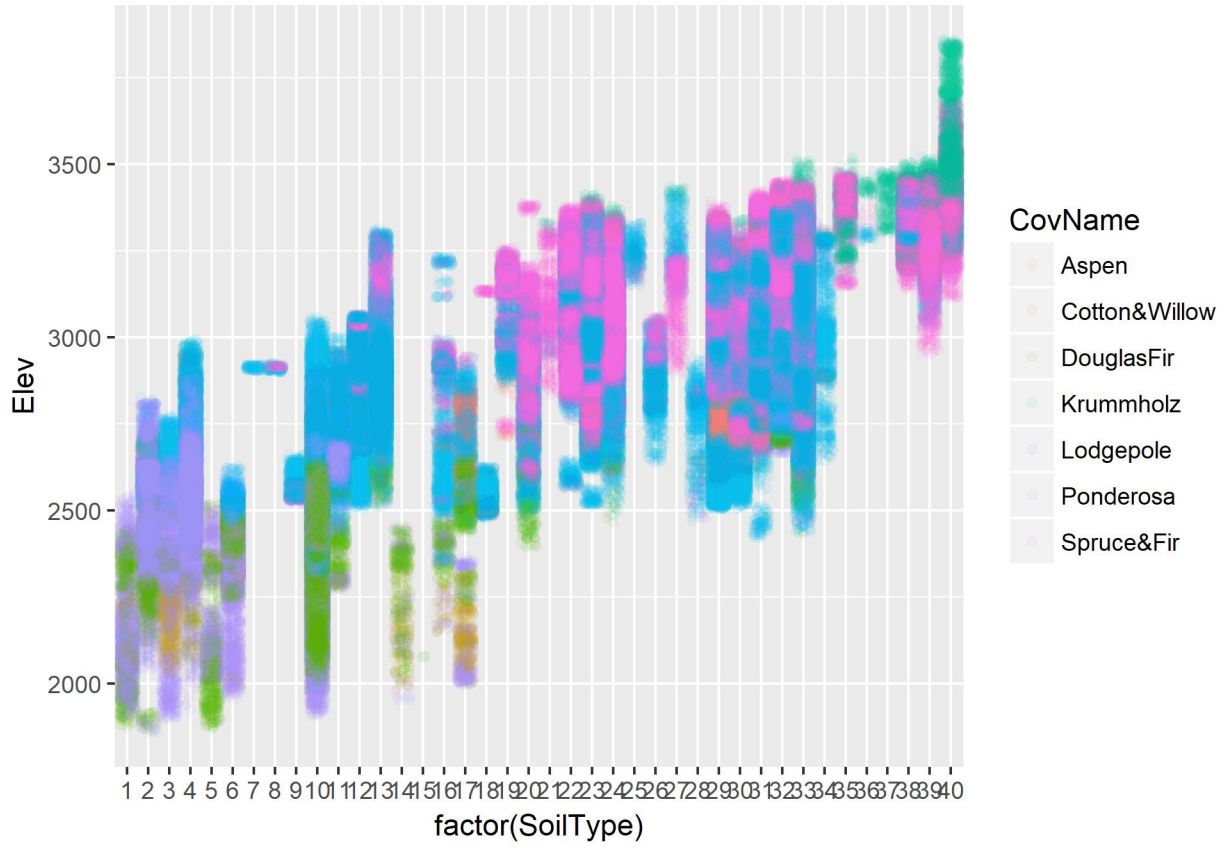

Figure 26: Elevation vs Aggregated Soil Type with Tree Type

Elevation vs Aggregated Soil Type with Tree Type - Figure 24

```
# Figure 24
g <- ggplot(forestcover,aes(factor(SoilType),Elev, col=CovName)) +
       geom_jitter(alpha=alphaVal)
  ggsave("Figure24.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

Seeing if Elevation vs Aggregated Soil Type gives any insights. Nothing jumping out at me.

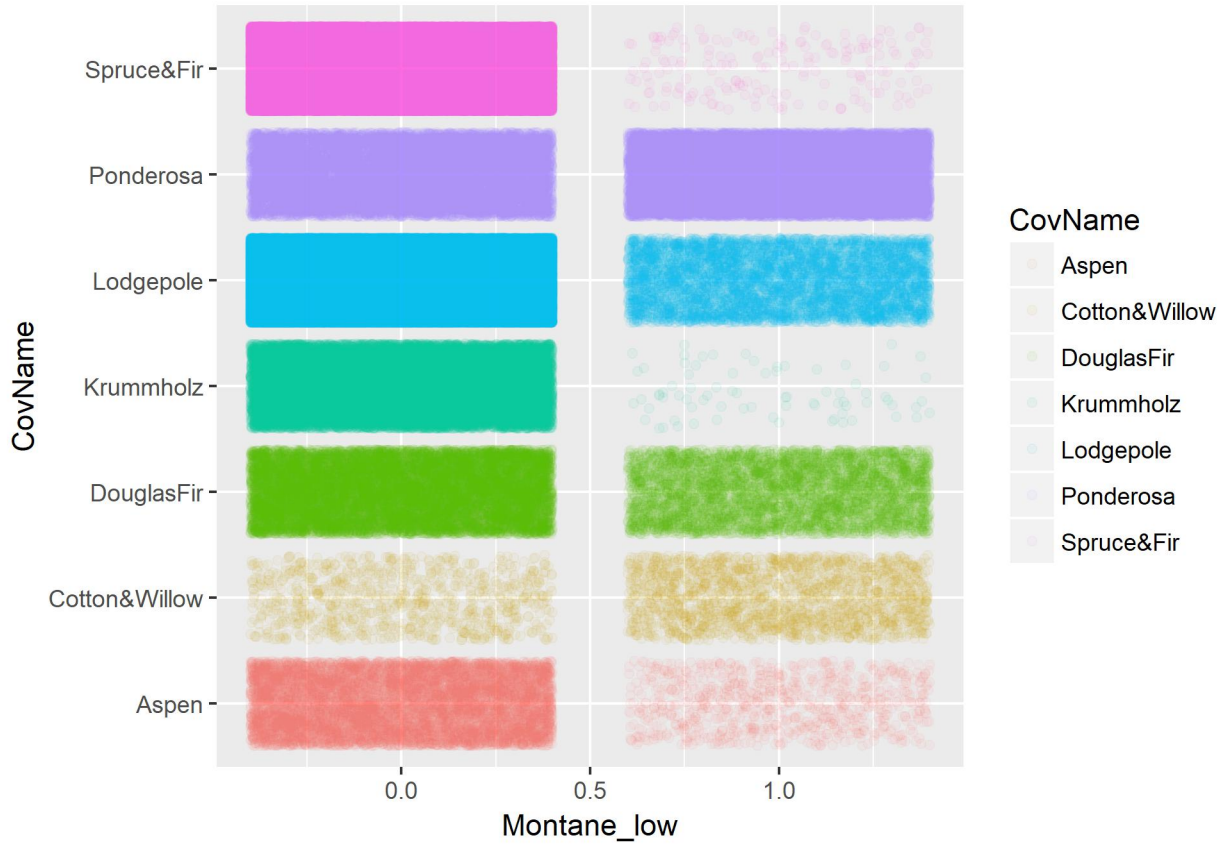

Figure 27: Tree Type vs "Montane Low" Soil Family

Tree Type vs "Montane Low" Soil Family - Figure 26

```
# Figure 26
g <- ggplot(forestcover,aes(Montane_low,CovName, col=CovName)) +
       geom_jitter(alpha=alphaVal)
  ggsave("Figure26.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

Trying to determine how best to look at the different Soil Families. Looking at an individual family is not very pretty. We really only care about the entries with Montane_low value of '1'.

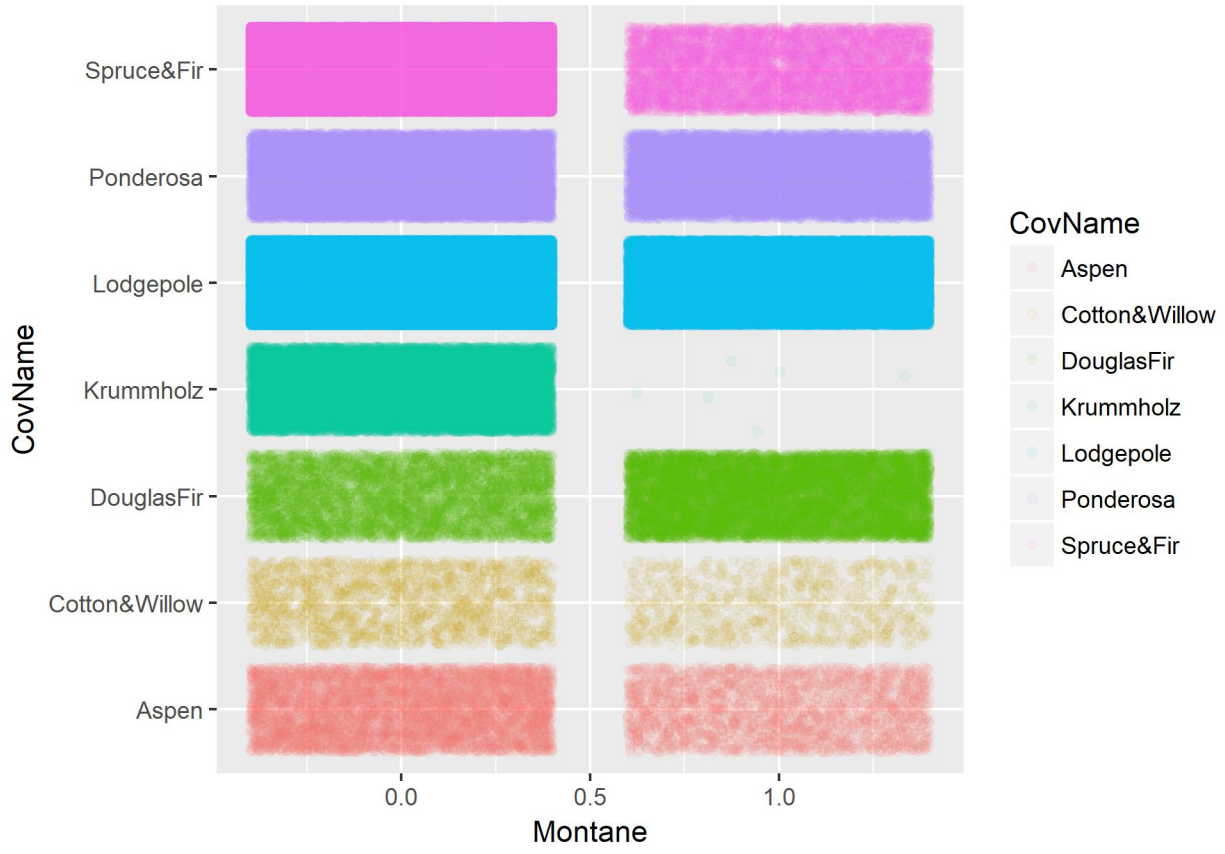

Figure 28: Tree Type vs "Montane" Soil Family

Tree Type vs "Montane" Soil Family - Figure 27

```
# Figure 27
g <- ggplot(forestcover,aes(Montane,CovName, col=CovName)) +
       geom_jitter(alpha=alphaVal)
  ggsave("Figure27.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

Looking at 'Montane' soil family. No conclusions here.

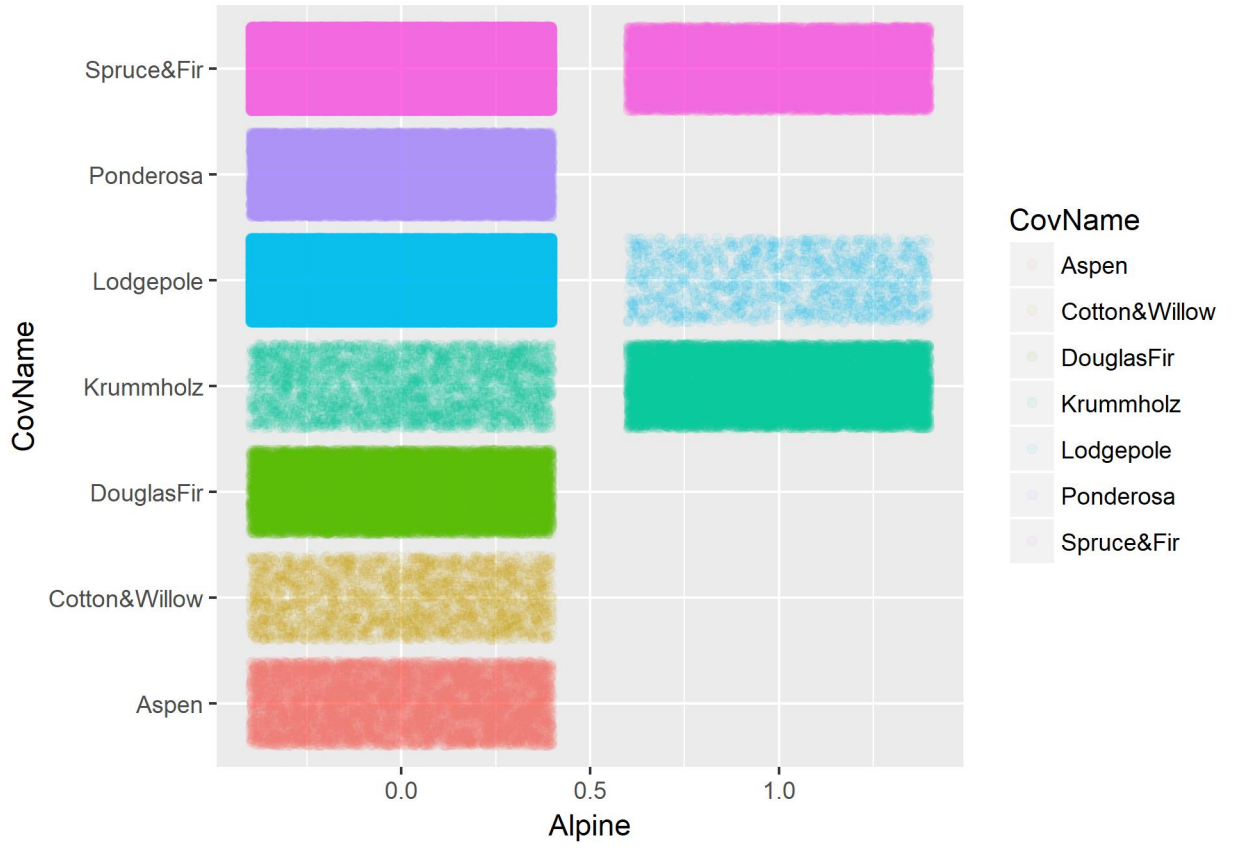

Figure 29: Tree Type vs "Alpine" Climate Zone

Tree Type vs "Alpine" Climate Zone - Figure 28

```
# Figure 28
g <- ggplot(forestcover,aes(Alpine,CovName, col=CovName)) +
       geom_jitter(alpha=alphaVal)
  ggsave("Figure28.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

The individual Alpine climate zone vs Tree cover. The Alpine Climate zone eliminates four of the tree types.

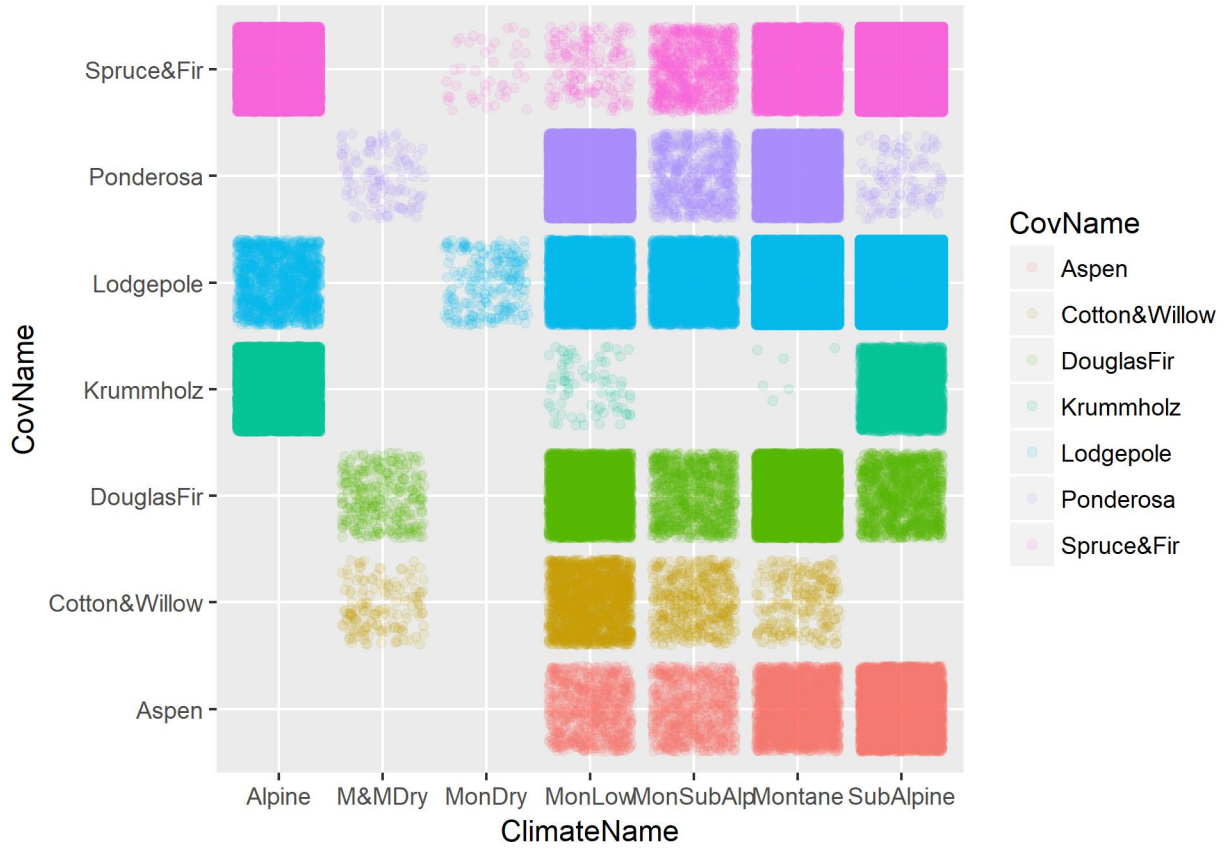

Figure 30: Tree Type vs Climate Zone

Tree Type vs Climate Zone - Figure 29

```
# Figure 29
g <- ggplot(forestcover,aes(ClimateName,CovName, col=CovName)) +
       geom_jitter(alpha=0.1)
  ggsave("Figure29.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

All of the Climate zones are graphed here. The Montane and Montane Dry (M&MDry) zone and the Montane Dry (MonDry) zones eliminate some tree types but the other zones are not as effective.

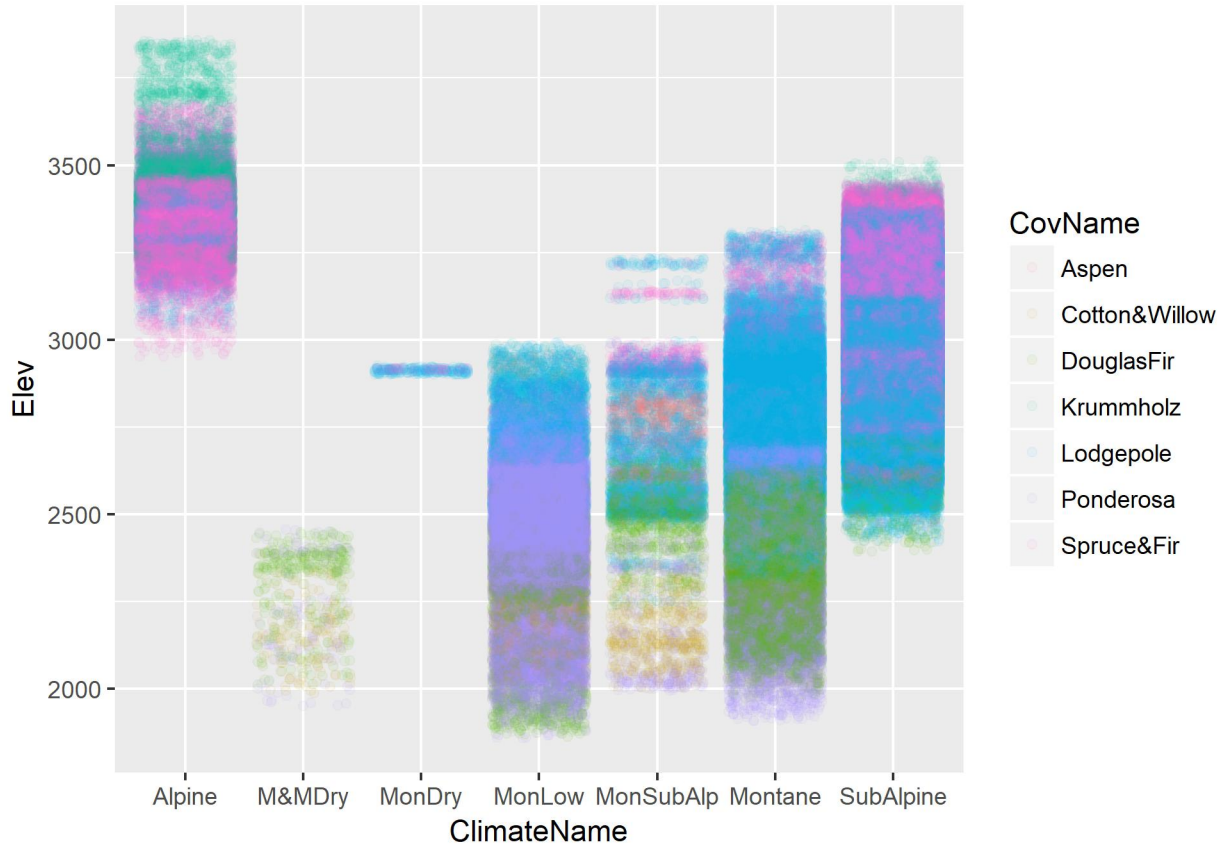

Figure 31: Elevation vs Climate with Tree Type

Elevation vs Climate with Tree Type - Figure 30

```
# Figure 30
g <- ggplot(forestcover,aes(ClimateName,Elev, col=CovName)) +
       geom_jitter(alpha=alphaVal)
  ggsave("Figure30.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

It's hard to see any patterns when looking at climate and Elevation with Tree type.

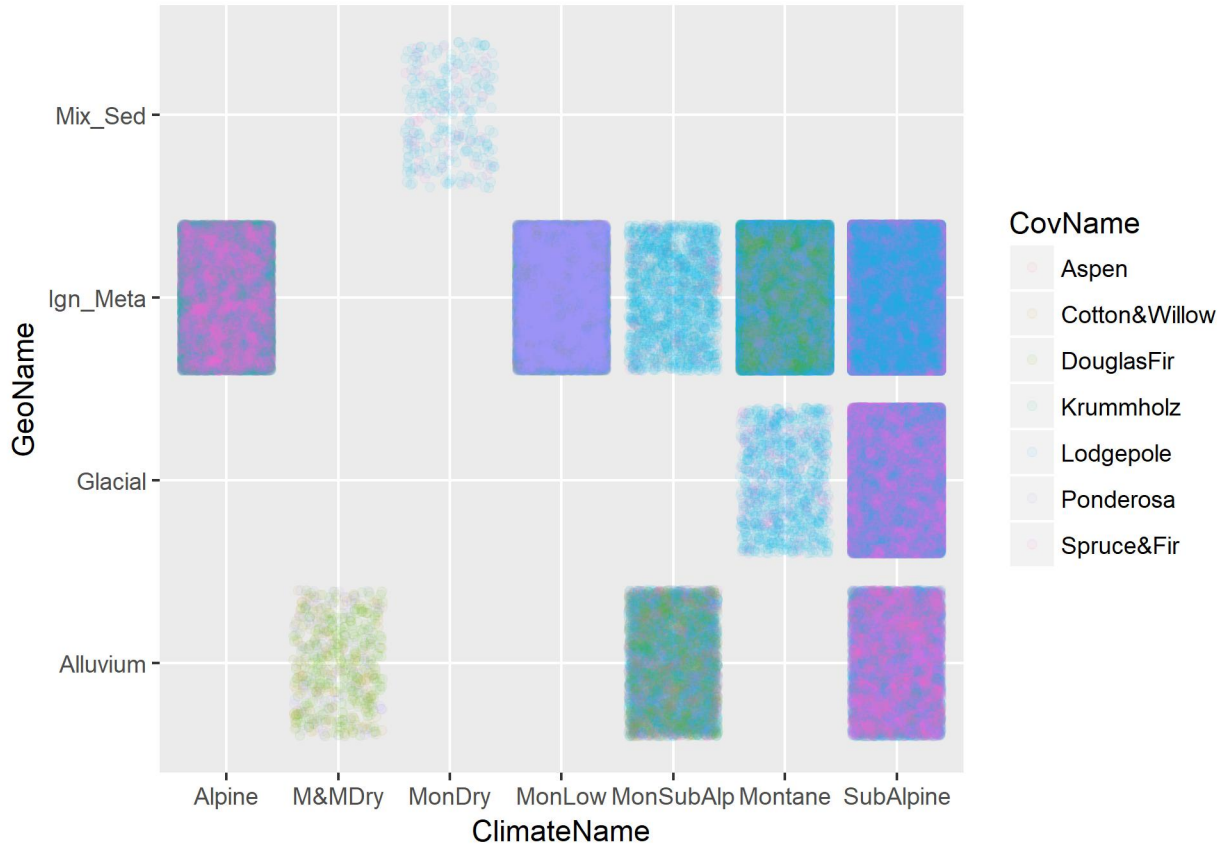

Figure 32: Geologic Zone vs Climate with Tree Type

Geologic Zone vs Climate with Tree Type - Figure 31

```
# Figure 31
g <- ggplot(forestcover,aes(ClimateName,GeoName, col=CovName)) +
       geom_jitter(alpha=alphaVal)
  ggsave("Figure31.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

Plotting Climate and Geologic zones with Tree Type brings in too much data to see any clear patterns but it looks like it would aid in classifying the data.

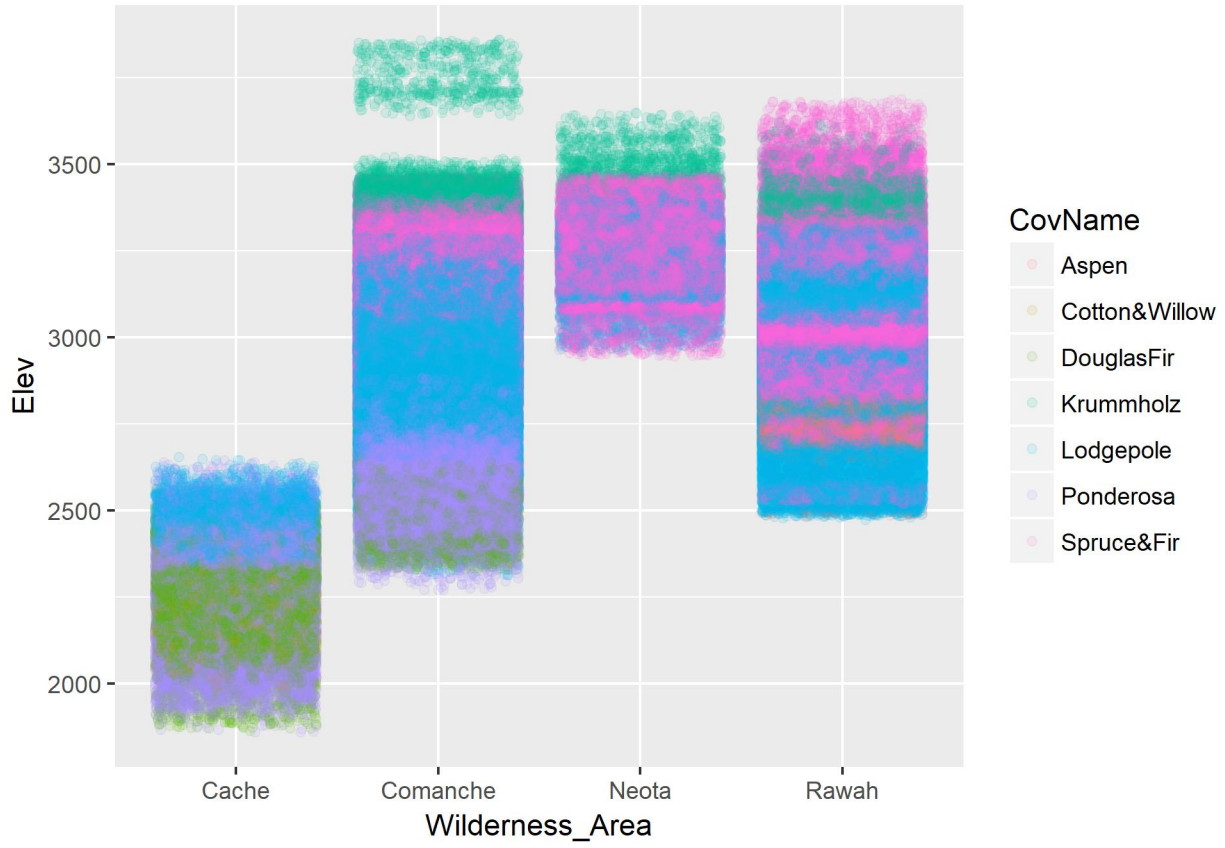

Figure 33: Elevation vs Wilderness Area with Tree Type

Elevation vs Wilderness Area with Tree Type - Figure 32

```
# Figure 32
g <- ggplot(forestcover,aes(Wilderness_Area,Elev,col=CovName)) +
        geom_jitter(alpha=0.1) # +
      #  facet_grid(. ~ CovName) +
  ggsave("Figure32.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

Elevation vs Wilderness area shows the wilderness area should be able to help classifying tree type.

Geology Count grouped Tree Type - Figure 33

```
# Figure 33
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------- tidyverse
```

```
## v tibble  1.4.2     v purrr   0.2.4
## v tidyr   0.8.0     v stringr 1.3.0
## v readr   1.1.1     v forcats 0.3.0
```

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

```
col1 <- grep("^CovName$", colnames(forestcover))
col2 <- grep("^Alluvium$", colnames(forestcover))
col3 <- grep("^Ign_Meta$", colnames(forestcover))
cols=c(col1,col2:col3)
#td2 <- gather(forestcover,Property,val,cols)
td2 <- forestcover[,cols]
td3<-gather(td2,Geology,Type,-1)
td4<- group_by(td3,CovName,Geology) %>%
  summarize(tot=sum(Type))
gr <- ggplot(td4, aes(CovName, tot, fill = Geology)) +
    geom_bar(stat = "identity", position = "dodge")
ggsave("Figure33.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

Plotting histograms of Tree Type grouped by Geologic Zone shows a similar shape between Geologic Zones.

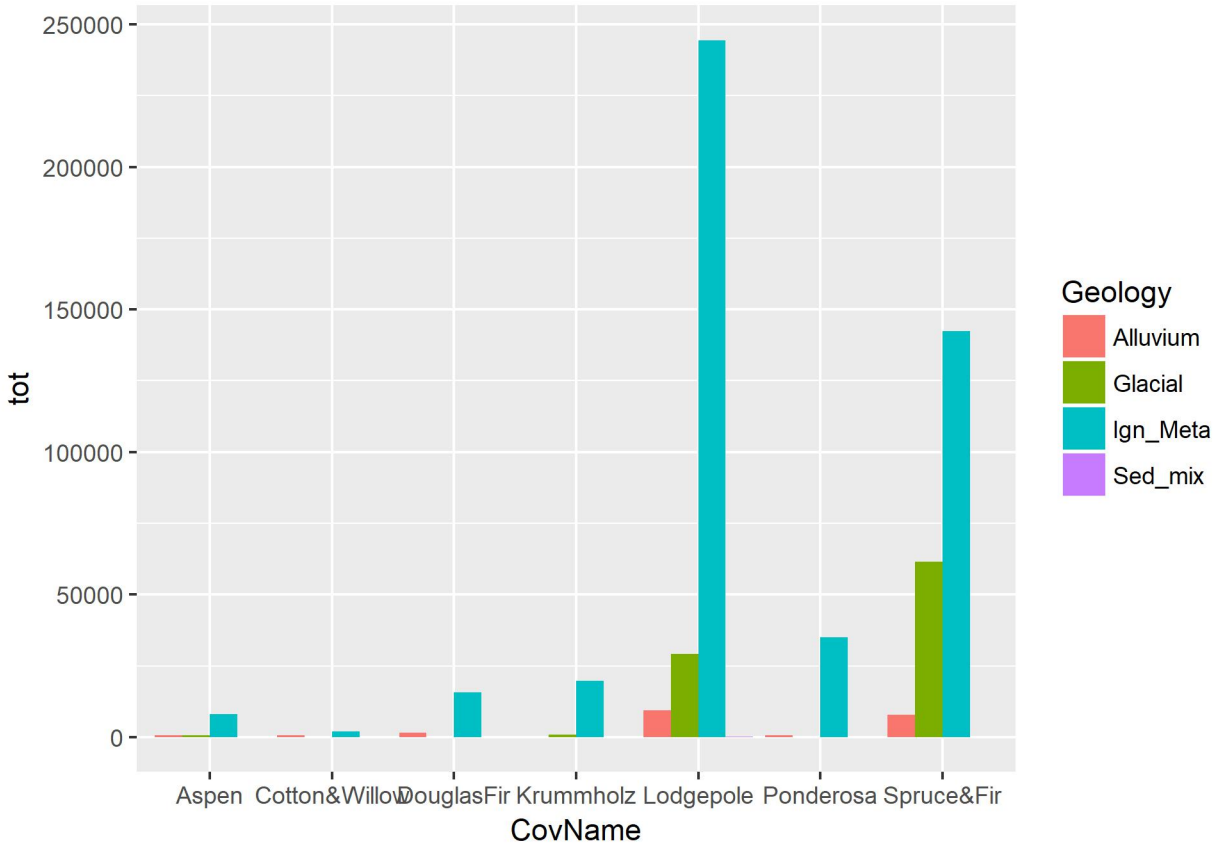

Figure 34: Geology Count grouped Tree Type

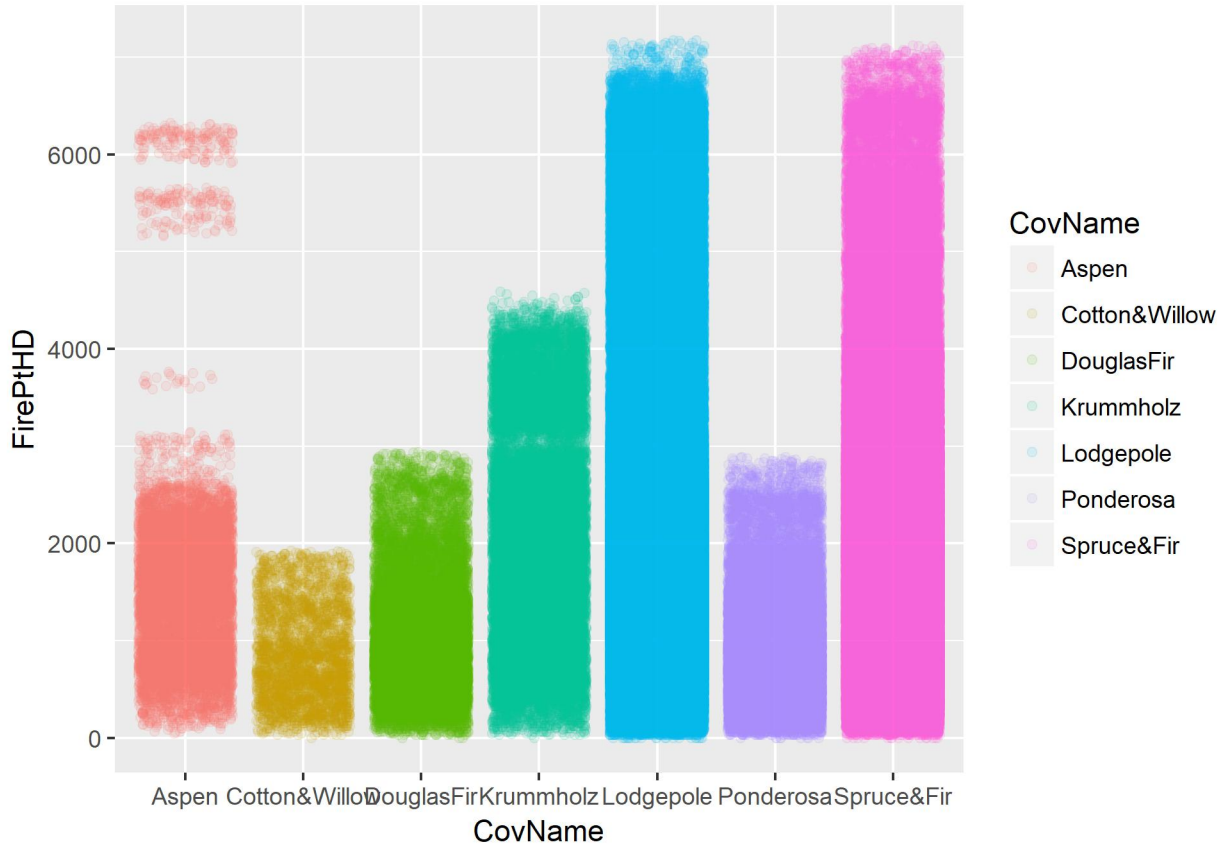

Figure 35: Fire Point Horizontal Distance vs Tree Type

Fire Point Horizontal Distance vs Tree Type - Figure 34

```
# Figure 34
  g <- ggplot(forestcover,aes(CovName, FirePtHD,col=CovName)) +
        geom_jitter(alpha=0.1) # +
      #  facet_grid(. ~ CovName) +
  ggsave("Figure34.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

It looks like the Fire Point distance can help aid in classifying the tree type by eliminating some tree types based on increasing distance.

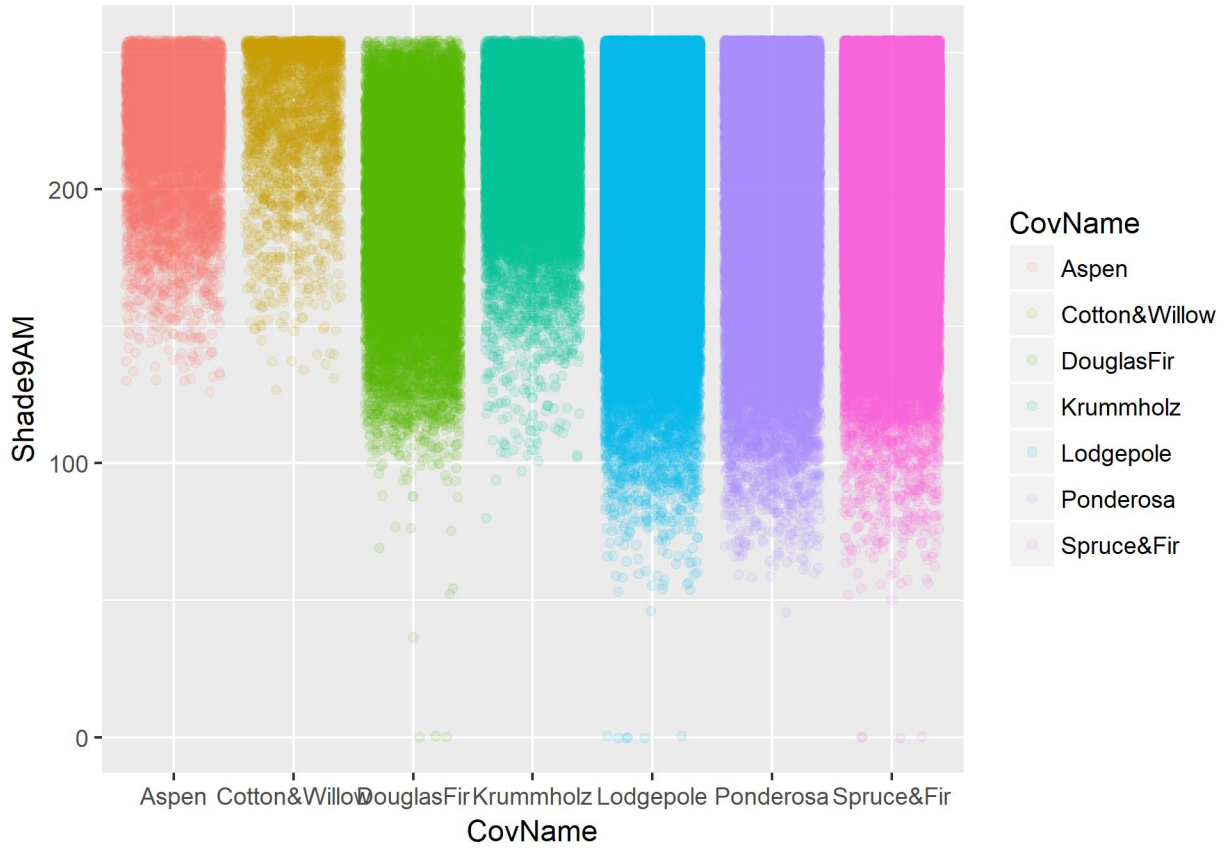

Figure 36: 9AM Shade vs Tree Type

9AM Shade vs Tree Type - Figure 35

```
# Figure 35
  g <- ggplot(forestcover,aes(CovName, Shade9AM,col=CovName)) +
        geom_jitter(alpha=0.1) # +
      #  facet_grid(. ~ CovName) +
  ggsave("Figure35.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

The Shade9AM value can be used to eliminate some trees based on low shade value.

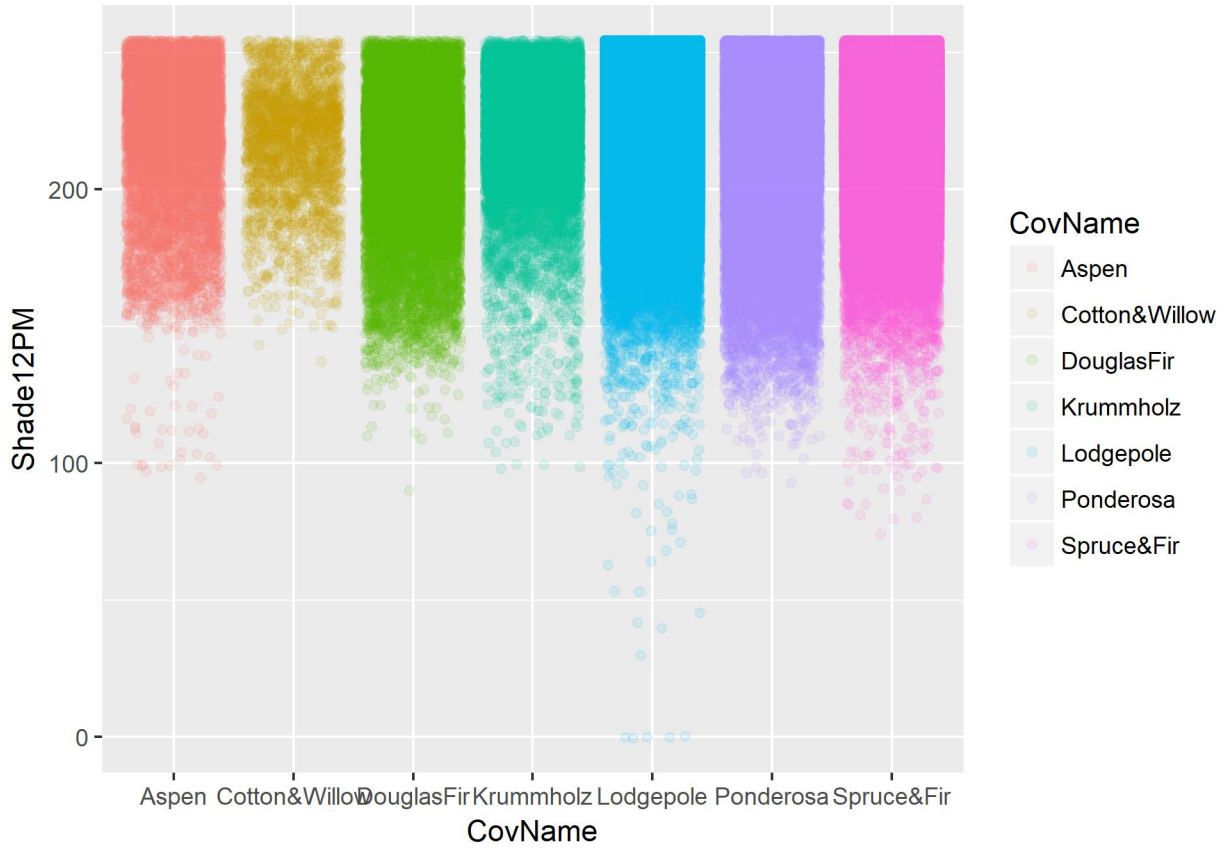

Figure 37: 12PM Shade vs Tree Type

12PM Shade vs Tree Type - Figure 36

```
# Figure 36
  g <- ggplot(forestcover,aes(CovName, Shade12PM,col=CovName)) +
        geom_jitter(alpha=0.1) # +
      #  facet_grid(. ~ CovName) +
  ggsave("Figure36.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

It looks like Shade12PM value can be used to help classify trees similar to Shade9AM but not as effective.

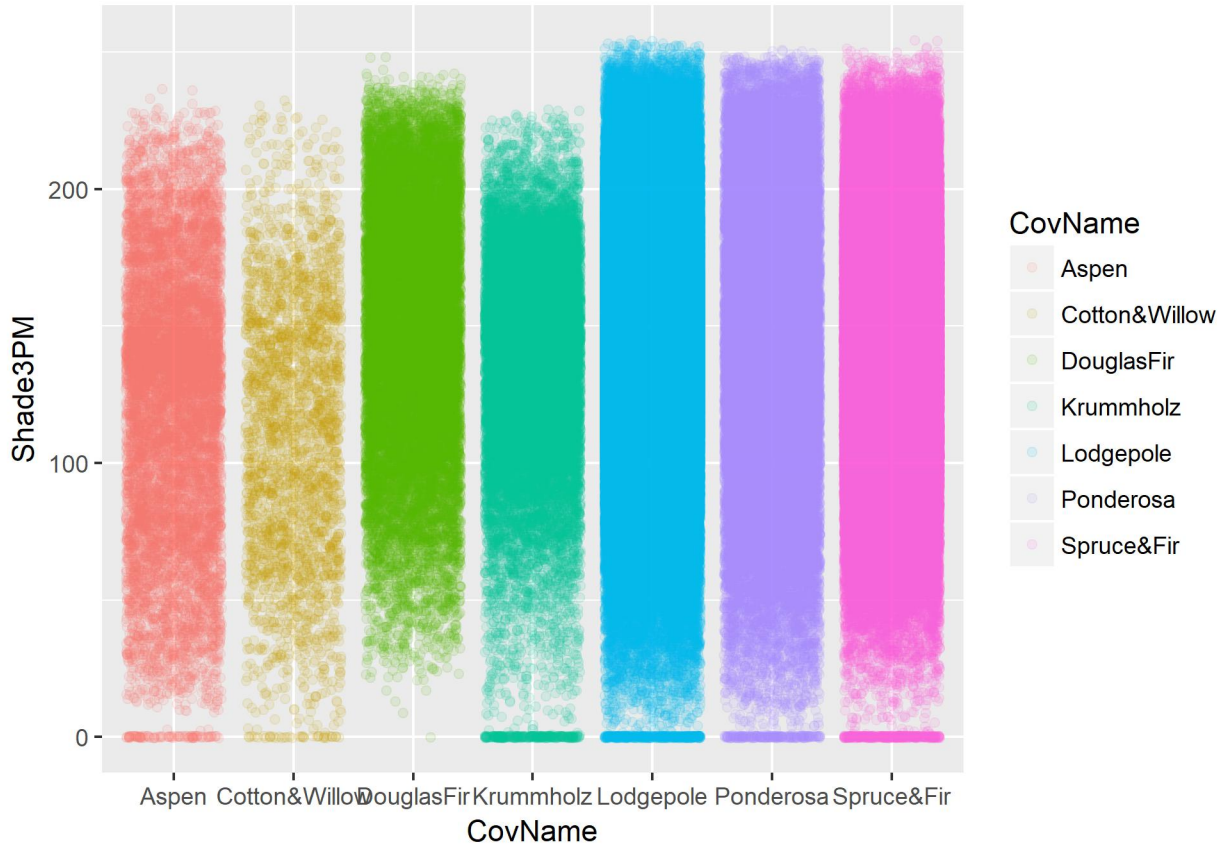

Figure 38: 3PM Shade vs Tree Type

3PM Shade vs Tree Type - Figure 37

```
# Figure 37
  g <- ggplot(forestcover,aes(CovName, Shade3PM,col=CovName)) +
        geom_jitter(alpha=0.1) # +
      #  facet_grid(. ~ CovName) +
  ggsave("Figure37.jpg")
```

```
## Saving 6.5 x 4.5 in image
```

The Shade3PM data does not look like it will help much to help classify tree type.

Data Analysis

After looking at some of the data relationships graphically, some statistical tests are applied to the data to test if variables follow a normal distribution or are independent.

Shaprio Test - Elevation

The Elevation histogram looked like it possibly had a normal distribution. It is not perfect but might be close enough statistically.

The Shapiro test is used to determine if data is normally distributed. The maximum number of data points for this Shaprio test is 5000. The alternate forest cover data extracted earlier is used. The Shapiro test result is shown below.

```
shapiro.test(altforestcover$Elev)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  altforestcover$Elev
## W = 0.95909, p-value < 2.2e-16
```

The null hypothesis for the Shapiro test is that the data follows a normal distribution. If the P-value is less than the 0.05 significance level, the null hypothesis is rejected and the data is not considered to be normally distributed otherwise the data is normally distributed.

The P-value for elevation data is 2e-16 which is nearly zero and much less than 0.05, therefore the null hypothesis is rejected and the data is not normally distributed. The previous histogram shows this visually: The graph has a longer left tail than right tail.

Chi Square Test

It looks like elevation can be used to help identify coverage type. A chi-square test will be used to see if the coverage type and elevation variables are independent.

To help with analysis, a function to calculate expected values for a contingency table is created as shown next.

```
# Create a table of expected values from a contigency table
expValues <- function(dFrame,debug) {
  wFrame<-dFrame              # create a new data frame with the same dimensions as passed in
  cstcsums<-colSums(wFrame)   # get a total count for each row
  cstrsums<-rowSums(wFrame)   # get a total count for each column
  csumTot<-sum(cstcsums)      # get a total count of all data points

  if (debug) {                # display the column and row counts if requested
    print(paste("colsums=",cstcsums))
    print(paste("rowsums=",cstrsums))
    print(paste("Total Count",csumTot))
  }

  # calculate the expected value for each cell
  minErr<-0                             # keep track of any errors (min expected value is 5 for chi-sqr
  for(i in 1:nrow(wFrame)) {
    for(j in 1:ncol(wFrame)) {
      expval<-as.integer(cstrsums[i]*10*cstcsums[j]/csumTot)/10  # truncate expected value to 0.1
      if(expval<5) {
        if(debug) {
          print(paste("Warn: Cell[",i,",",j,"]=",expval," is less than 5!"))
        }
        minErr<-minErr+1
      }
      wFrame[i,j]<-expval     # save the expected value in the table
    }
  }
  if (minErr > 0) {
    print(paste("WARNING! There were",minErr,"cells with expected values less than 5."))
  }
  wFrame  # return the table
}
```

Now, create a contigency table for Coverage Type vs Elevation Bin, print the table and check that the sum of the counts add up to the number of rows in the forest coverage data.

```
chisqtbl<-table(forestcover$CovName, forestcover$ElevSlot)
chisqtbl
```

```
##
##                   18    19    20    21    22    23    24    25    26
##   Aspen            0     0     0     0     0     0    66   465   808
##   Cotton&Willow    0     9   356   715   940   664    59     4     0
##   DouglasFir      68   441   646  1080  1623  3468  4323  2744  1861
##   Krummholz        0     0     0     0     0     0     0     0     0
##   Lodgepole        0     0     0    45   209   421  1931 12527 22418
##   Ponderosa       23   717  2425  3394  4428  6575  6423  5978  4152
##   Spruce&Fir       0     0     0     0     0     0     9   488  1525
##
```

```
##                    27    28    29    30    31    32    33    34    35
##   Aspen          3576  3523  1049     6     0     0     0     0     0
##   Cotton&Willow     0     0     0     0     0     0     0     0     0
##   DouglasFir      916   196     1     0     0     0     0     0     0
##   Krummholz         0    24    60     0   594  4450  9260  4884   604
##   Lodgepole     35423 45902 68630 47410 28490 16979  2838    78     0
##   Ponderosa      1339   300     0     0     0     0     0     0     0
##   Spruce&Fir     5021 10350 23667 40965 55028 50325 19046  4114  1021
##
##                    36    37    38
##   Aspen             0     0     0
##   Cotton&Willow     0     0     0
##   DouglasFir        0     0     0
##   Krummholz       193   285   156
##   Lodgepole         0     0     0
##   Ponderosa         0     0     0
##   Spruce&Fir      281     0     0
```

```
sum(colSums(chisqtbl))
```

```
## [1] 581012
```

Create an expected values table. We want to ensure that the minimum expected value for each cell is at least 5, otherwise the chi-square test is not valid.

```
expVals<-expValues(chisqtbl,1)
```

```
##  [1] "colsums= 91"    "colsums= 1167"  "colsums= 3427"  "colsums= 5234"
##  [5] "colsums= 7200"  "colsums= 11128" "colsums= 12811" "colsums= 22206"
##  [9] "colsums= 30764" "colsums= 46275" "colsums= 60295" "colsums= 93407"
## [13] "colsums= 88381" "colsums= 84112" "colsums= 71754" "colsums= 31144"
## [17] "colsums= 9076"  "colsums= 1625"  "colsums= 474"   "colsums= 285"
## [21] "colsums= 156"
## [1] "rowsums= 9493"   "rowsums= 2747"   "rowsums= 17367"  "rowsums= 20510"
## [5] "rowsums= 283301" "rowsums= 35754"  "rowsums= 211840"
## [1] "Total Count 581012"
## [1] "Warn: Cell[ 1 , 1 ]= 1.4  is less than 5!"
## [1] "Warn: Cell[ 1 , 20 ]= 4.6  is less than 5!"
## [1] "Warn: Cell[ 1 , 21 ]= 2.5  is less than 5!"
## [1] "Warn: Cell[ 2 , 1 ]= 0.4  is less than 5!"
## [1] "Warn: Cell[ 2 , 19 ]= 2.2  is less than 5!"
## [1] "Warn: Cell[ 2 , 20 ]= 1.3  is less than 5!"
## [1] "Warn: Cell[ 2 , 21 ]= 0.7  is less than 5!"
## [1] "Warn: Cell[ 3 , 1 ]= 2.7  is less than 5!"
## [1] "Warn: Cell[ 3 , 21 ]= 4.6  is less than 5!"
## [1] "Warn: Cell[ 4 , 1 ]= 3.2  is less than 5!"
## [1] "WARNING! There were 10 cells with expected values less than 5."
```

```
expVals
```

```
##
##                    18      19      20      21      22      23      24
##   Aspen            1.4    19.0    55.9    85.5   117.6   181.8   209.3
##   Cotton&Willow    0.4     5.5    16.2    24.7    34.0    52.6    60.5
##   DouglasFir       2.7    34.8   102.4   156.4   215.2   332.6   382.9
##   Krummholz        3.2    41.1   120.9   184.7   254.1   392.8   452.2
##   Lodgepole       44.3   569.0  1671.0  2552.0  3510.7  5426.0  6246.6
```

```
##   Ponderosa         5.5    71.8   210.8   322.0   443.0   684.7   788.3
##   Spruce&Fir       33.1   425.4  1249.5  1908.3  2625.1  4057.3  4670.9
##
##                      25      26      27      28      29      30      31
##   Aspen           362.8   502.6   756.0   985.1  1526.1  1444.0  1374.2
##   Cotton&Willow   104.9   145.4   218.7   285.0   441.6   417.8   397.6
##   DouglasFir      663.7   919.5  1383.2  1802.2  2792.0  2641.7  2514.1
##   Krummholz       783.8  1085.9  1633.5  2128.4  3297.3  3119.8  2969.1
##   Lodgepole     10827.6 15000.5 22563.6 29399.7 45545.1 43094.5 41012.9
##   Ponderosa      1366.5  1893.1  2847.6  3710.4  5748.0  5438.7  5176.0
##   Spruce&Fir     8096.4 11216.7 16872.1 21983.8 34056.6 32224.1 30667.6
##
##                      32      33      34      35      36      37      38
##   Aspen          1172.3   508.8   148.2    26.5     7.7     4.6     2.5
##   Cotton&Willow   339.2   147.2    42.9     7.6     2.2     1.3     0.7
##   DouglasFir     2144.7   930.9   271.2    48.5    14.1     8.5     4.6
##   Krummholz      2532.9  1099.3   320.3    57.3    16.7    10.0     5.5
##   Lodgepole     34987.1 15185.7  4425.4   792.3   231.1   138.9    76.0
##   Ponderosa      4415.5  1916.5   558.5    99.9    29.1    17.5     9.5
##   Spruce&Fir    26161.8 11355.2  3309.1   592.4   172.8   103.9    56.8
```

```
sum(colSums(expVals))
```

```
## [1] 581004.4
```

Here we see there are several cells with expected values less than 5. The chi-square test may not be valid and the chi-square test itself (below) says the test may not be valid. Although we don't know if expected counts less than 5 is the reason for the message.

```
cst <- chisq.test(chisqtbl, correct = FALSE)
```

```
## Warning in chisq.test(chisqtbl, correct = FALSE): Chi-squared approximation
## may be incorrect
```

```
cst
```

```
##
##  Pearson's Chi-squared test
##
## data:  chisqtbl
## X-squared = 763760, df = 120, p-value < 2.2e-16
```

If the P-value is less than significance factor of 0.05, the null hypothesis is rejected and the variables are not independent. The P-value is 2e-16 which is nearly zero. This shows that the coverage type and elevation are dependent, if the chi-square test is valid.

Coverage Type vs Soil Type Independence check

The original paper used the Soil Type categories to predict coverage type. Let's try a chi-square test on them. The contigency table is created, printed and total counts listed.

```
chisqtbl<-table(forestcover$CovName, forestcover$SoilType)
chisqtbl
```

```
##
##                   1     2     3     4     5     6     7     8     9
##   Aspen           0   264     0   585     0     0     0     0     0
```

```
##   Cotton&Willow   178   115  1018   168    48   320     0     0     0
##   DouglasFir      752  1303   203   631   582  1350     0     0     0
##   Krummholz         0     0     0    78     0     0     0     0     0
##   Lodgepole         0   852  1191  3251     0   912   105   136   986
##   Ponderosa      2101  4991  2411  7501   967  3993     0     0     0
##   Spruce&Fir        0     0     0   182     0     0     0    43   161
##
##                    10    11    12    13    14    15    16    17    18
##   Aspen           260   681     0  1315     0     0    35   600   170
##   Cotton&Willow   224    34     0     0   155     0    51   436     0
##   DouglasFir     8859   518     0   614   328     3   251   709     0
##   Krummholz         0     0     0     6     0     0     0     0     0
##   Lodgepole     10803  9077 27278 13258     0     0  1743   957  1659
##   Ponderosa     11532  1353     0    41   116     0   129   506     0
##   Spruce&Fir      956   747  2693  2197     0     0   636   214    70
##
##                    19    20    21    22    23    24    25    26    27
##   Aspen            67    53     0     0   699    70     0   132     0
##   Cotton&Willow     0     0     0     0     0     0     0     0     0
##   DouglasFir        0   280     0     0    29   138     0     0     0
##   Krummholz         3     0    13   148   706   204     0     0    31
##   Lodgepole      1490  5207    21  7442 20761  9702   349  2174   451
##   Ponderosa         0     2     0     0     0     0     0     0     0
##   Spruce&Fir     2461  3717   804 25783 35557 11164   125   283   604
##
##                    28    29    30    31    32    33    34    35    36
##   Aspen            12  1132  2111   309   460   518    20     0     0
##   Cotton&Willow     0     0     0     0     0     0     0     0     0
##   DouglasFir        0     0     0    63   200   539    15     0     0
##   Krummholz         0   805   197   222   839   636    51   948    63
##   Lodgepole       891 71399 20218 13209 29556 25308  1431    12    42
##   Ponderosa         0     0     0     0   106     5     0     0     0
##   Spruce&Fir       43 41911  7644 11863 21358 18148    94   931    14
##
##                    37    38    39    40
##   Aspen             0     0     0     0
##   Cotton&Willow     0     0     0     0
##   DouglasFir        0     0     0     0
##   Krummholz       298  6104  5566  3592
##   Lodgepole         0   740   358   332
##   Ponderosa         0     0     0     0
##   Spruce&Fir        0  8729  7882  4826
```

```
sum(colSums(chisqtbl))
```

```
## [1] 581012
```

Next an expected value table is created to verify that all expected values are at least 5.

```
expVals<-expValues(chisqtbl,1)
```

```
##  [1] "colsums= 3031"   "colsums= 7525"   "colsums= 4823"
##  [4] "colsums= 12396"  "colsums= 1597"   "colsums= 6575"
##  [7] "colsums= 105"    "colsums= 179"    "colsums= 1147"
## [10] "colsums= 32634"  "colsums= 12410"  "colsums= 29971"
## [13] "colsums= 17431"  "colsums= 599"    "colsums= 3"
```

```
## [16] "colsums= 2845"   "colsums= 3422"   "colsums= 1899"
## [19] "colsums= 4021"   "colsums= 9259"   "colsums= 838"
## [22] "colsums= 33373"  "colsums= 57752"  "colsums= 21278"
## [25] "colsums= 474"    "colsums= 2589"   "colsums= 1086"
## [28] "colsums= 946"    "colsums= 115247" "colsums= 30170"
## [31] "colsums= 25666"  "colsums= 52519"  "colsums= 45154"
## [34] "colsums= 1611"   "colsums= 1891"   "colsums= 119"
## [37] "colsums= 298"    "colsums= 15573"  "colsums= 13806"
## [40] "colsums= 8750"
## [1] "rowsums= 9493"   "rowsums= 2747"   "rowsums= 17367"  "rowsums= 20510"
## [5] "rowsums= 283301" "rowsums= 35754"  "rowsums= 211840"
## [1] "Total Count 581012"
## [1] "Warn: Cell[ 1 , 7 ]= 1.7  is less than 5!"
## [1] "Warn: Cell[ 1 , 8 ]= 2.9  is less than 5!"
## [1] "Warn: Cell[ 1 , 15 ]= 0  is less than 5!"
## [1] "Warn: Cell[ 1 , 36 ]= 1.9  is less than 5!"
## [1] "Warn: Cell[ 1 , 37 ]= 4.8  is less than 5!"
## [1] "Warn: Cell[ 2 , 7 ]= 0.4  is less than 5!"
## [1] "Warn: Cell[ 2 , 8 ]= 0.8  is less than 5!"
## [1] "Warn: Cell[ 2 , 14 ]= 2.8  is less than 5!"
## [1] "Warn: Cell[ 2 , 15 ]= 0  is less than 5!"
## [1] "Warn: Cell[ 2 , 21 ]= 3.9  is less than 5!"
## [1] "Warn: Cell[ 2 , 25 ]= 2.2  is less than 5!"
## [1] "Warn: Cell[ 2 , 28 ]= 4.4  is less than 5!"
## [1] "Warn: Cell[ 2 , 36 ]= 0.5  is less than 5!"
## [1] "Warn: Cell[ 2 , 37 ]= 1.4  is less than 5!"
## [1] "Warn: Cell[ 3 , 7 ]= 3.1  is less than 5!"
## [1] "Warn: Cell[ 3 , 15 ]= 0  is less than 5!"
## [1] "Warn: Cell[ 3 , 36 ]= 3.5  is less than 5!"
## [1] "Warn: Cell[ 4 , 7 ]= 3.7  is less than 5!"
## [1] "Warn: Cell[ 4 , 15 ]= 0.1  is less than 5!"
## [1] "Warn: Cell[ 4 , 36 ]= 4.2  is less than 5!"
## [1] "Warn: Cell[ 5 , 15 ]= 1.4  is less than 5!"
## [1] "Warn: Cell[ 6 , 15 ]= 0.1  is less than 5!"
## [1] "Warn: Cell[ 7 , 15 ]= 1  is less than 5!"
## [1] "WARNING! There were 23 cells with expected values less than 5."
```

```
expVals
```

```
##
##                      1       2       3       4       5       6       7
##   Aspen            49.5   122.9    78.8   202.5    26.0   107.4     1.7
##   Cotton&Willow    14.3    35.5    22.8    58.6     7.5    31.0     0.4
##   DouglasFir       90.5   224.9   144.1   370.5    47.7   196.5     3.1
##   Krummholz       106.9   265.6   170.2   437.5    56.3   232.1     3.7
##   Lodgepole      1477.9  3669.1  2351.6  6044.2   778.6  3205.9    51.1
##   Ponderosa       186.5   463.0   296.7   762.8    98.2   404.6     6.4
##   Spruce&Fir     1105.1  2743.6  1758.4  4519.6   582.2  2397.2    38.2
##
##                      8       9      10      11      12      13      14
##   Aspen             2.9    18.7   533.1   202.7   489.6   284.8     9.7
##   Cotton&Willow     0.8     5.4   154.2    58.6   141.7    82.4     2.8
##   DouglasFir        5.3    34.2   975.4   370.9   895.8   521.0    17.9
##   Krummholz         6.3    40.4  1151.9   438.0  1057.9   615.3    21.1
##   Lodgepole        87.2   559.2 15912.3  6051.1 14613.8  8499.3   292.0
```

```
##   Ponderosa        11.0    70.5  2008.2   763.6  1844.3  1072.6    36.8
##   Spruce&Fir       65.2   418.2 11898.5  4524.7 10927.5  6355.4   218.3
##
##                      15      16      17      18      19      20      21
##   Aspen             0.0    46.4    55.9    31.0    65.6   151.2    13.6
##   Cotton&Willow     0.0    13.4    16.1     8.9    19.0    43.7     3.9
##   DouglasFir        0.0    85.0   102.2    56.7   120.1   276.7    25.0
##   Krummholz         0.1   100.4   120.7    67.0   141.9   326.8    29.5
##   Lodgepole         1.4  1387.2  1668.5   925.9  1960.6  4514.6   408.6
##   Ponderosa         0.1   175.0   210.5   116.8   247.4   569.7    51.5
##   Spruce&Fir        1.0  1037.3  1247.6   692.3  1466.0  3375.8   305.5
##
##                      22      23      24      25      26      27      28
##   Aspen           545.2   943.5   347.6     7.7    42.3    17.7    15.4
##   Cotton&Willow   157.7   273.0   100.6     2.2    12.2     5.1     4.4
##   DouglasFir      997.5  1726.2   636.0    14.1    77.3    32.4    28.2
##   Krummholz      1178.0  2038.6   751.1    16.7    91.3    38.3    33.3
##   Lodgepole     16272.6 28159.8 10375.1   231.1  1262.3   529.5   461.2
##   Ponderosa      2053.6  3553.9  1309.3    29.1   159.3    66.8    58.2
##   Spruce&Fir    12167.9 21056.6  7758.0   172.8   943.9   395.9   344.9
##
##                      29      30      31      32      33      34      35
##   Aspen          1882.9   492.9   419.3   858.0   737.7    26.3    30.8
##   Cotton&Willow   544.8   142.6   121.3   248.3   213.4     7.6     8.9
##   DouglasFir     3444.8   901.8   767.1  1569.8  1349.6    48.1    56.5
##   Krummholz      4068.2  1065.0   906.0  1853.9  1593.9    56.8    66.7
##   Lodgepole     56194.3 14710.8 12514.7 25608.2 22017.0   785.5   922.0
##   Ponderosa      7092.0  1856.5  1579.4  3231.8  2778.6    99.1   116.3
##   Spruce&Fir    42019.6 11000.1  9357.9 19148.7 16463.3   587.3   689.4
##
##                      36      37      38      39      40
##   Aspen             1.9     4.8   254.4   225.5   142.9
##   Cotton&Willow     0.5     1.4    73.6    65.2    41.3
##   DouglasFir        3.5     8.9   465.4   412.6   261.5
##   Krummholz         4.2    10.5   549.7   487.3   308.8
##   Lodgepole        58.0   145.3  7593.3  6731.7  4266.4
##   Ponderosa         7.3    18.3   958.3   849.5   538.4
##   Spruce&Fir       43.3   108.6  5677.9  5033.7  3190.2
```

```
sum(colSums(expVals))
```

```
## [1] 580997
```

Here we see there are several cells with expected values less than 5. The chi-square test may not be valid and the chi-square test itself (below) says the test may not be valid. Although we don't know if expected counts less than 5 is the reason for the message.

```
cst <- chisq.test(chisqtbl, correct = FALSE)
```

```
## Warning in chisq.test(chisqtbl, correct = FALSE): Chi-squared approximation
## may be incorrect
```

```
cst
```

```
##
##  Pearson's Chi-squared test
##
```

```
## data:  chisqtbl
## X-squared = 762250, df = 234, p-value < 2.2e-16
```

If the P-value is less than significance factor of 0.05, the null hypothesis is rejected and the variables are not independent. The P-value is 2e-16 which is nearly zero. This shows that the coverage type and soil type are dependent, if the chi-square test is valid.

Conclusion

There are many interesting data distributions in the continuous data. The elevation data seems to be the most easy to intuitively see a relationship predicting the outcome of the coverage type.

The other categorical data seems difficult to relate to outcome intuitively. More chi-square testing can be performed before this report is complete.

```
  endTime=Sys.time()
  print(paste("Figures completed at",endTime))
```

```
## [1] "Figures completed at 2018-06-07 06:57:45"
```

```
  print(paste("Elapsed time=",round(endTime-startTime),"seconds."))
```

```
## [1] "Elapsed time= 30 seconds."
```

```
progEnd=Sys.time()
print(paste("R script  started  at",progStart))
```

```
## [1] "R script  started  at 2018-06-07 06:27:25"
```

```
print(paste("R script completed at",progEnd))
```

```
## [1] "R script completed at 2018-06-07 06:57:45"
```

```
#print(paste("Elapsed time=",progEnd-progStart,"seconds."))
```

This concludes the current data exploration on my capstone data.