Capstone Data Logistic Regression - Predict Lodgepole Pine

Tom Thorpe

July 25, 2018

Objective

Use Logistic regression to predict tree coverage.

```
# Include required libraries.

library(gsubfn)
```

```
## Loading required package: proto
```

```
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
library(ROCR)
```

```
## Loading required package: gplots

##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess
```

```
suppressMessages(library(latticeExtra, warn.conflicts = FALSE, quietly=TRUE))
#library(latticeExtra)

  curTime=Sys.time()
  print(paste("Forest Cover Logistic script started at",curTime))
```

```
## [1] "Forest Cover Logistic script started at 2018-08-12 20:53:58"
```

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

calcROC <- 1
saveFileName="ForestCoverLogisticStats.csv"

infile="C:/Users/Tom/git/datasciencefoundation/ForestCoverage/forestcover_clean_full.csv"
```

```r
#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
```

```r
forestcover <- read.csv(infile,header=TRUE,sep=",") %>% tbl_df()
  curTime=Sys.time()
  print(paste("Forest Cover data load completed at",curTime))
```

```
## [1] "Forest Cover data load completed at 2018-08-12 20:54:38"
```

```r
forestcover$SoilType<-as.factor(forestcover$SoilType)
forestcover$ClimateZone<-as.factor(forestcover$ClimateZone)
forestcover$GeoZone<-as.factor(forestcover$GeoZone)

# glimpse(forestcover)

# table(forestcover$Sed_mix)
#knitr::knit_exit()

# Coverage binary outcome Vars:
# Aspen
# Cottonwood_Willow
# DouglasFir
# Krummholz
# LodgepolePine
# PonderosaPine
# Spruce_Fir
```

A table showing the number of occurrences for each tree type is shown below.

```r
covCount<-data.frame(table(forestcover$CovName))
totCount<-nrow(forestcover)
covCount <- mutate(covCount,Percent = as.integer(covCount$Freq*1000/totCount)/10)
LodgePct<-covCount$Percent[covCount$Var1=="Lodgepole"]
SpruceAndFirPct<-covCount$Percent[covCount$Var1=="Spruce&Fir"]
LodgeAndSpruceAndFir<-LodgePct+SpruceAndFirPct
#```
#```{r echo=TRUE}
covCount
```

```
##            Var1   Freq Percent
## 1         Aspen   9493     1.6
## 2 Cotton&Willow   2747     0.4
## 3    DouglasFir  17367     2.9
## 4     Krummholz  20510     3.5
## 5     Lodgepole 283301    48.7
## 6     Ponderosa  35754     6.1
## 7    Spruce&Fir 211840    36.4
```

Lodge pole Pine represents 48.7 percent of the sample. So always guessing "Lodge pole" would provide success rate of 48.7 percent and can be used as a baseline for comparing our predictions. Spruce & Fir represent the next largest number of trees. The two together represent 85.1 percent.

Logistic Model Accuracy Function

A function to help determine threshold for best accuracy and testing is shown below.

```
source("logisticAccuracy.R") # for calcLogisticModelAccuracy function
bestThreshIndex=11
#save("calcLogisticModelAccuracy", file="logisticAccuracy.Rdata")
```

Create Training and Testing Sets

Split data into training and testing data for logistic regression. The split is based on cover type so that the different coverage types will be split proportionately for all cover types in the training and test sets.

```
library(caTools)
set.seed(127)
split = sample.split(forestcover$CovType, 0.70) # we want 65% in the training set
forestTrain = subset(forestcover, split == TRUE)
forestTest  = subset(forestcover, split == FALSE)
```

Check training set coverage percentages and compare with test set to ensure there is a representative amount of data in each set for each coverage type.

View Training Set Coverage Percentages

Check training set coverage percentages.

```
covCount<-data.frame(table(forestTrain$CovName))
totCount<-nrow(forestTrain)
covCount <- mutate(covCount,Percent = as.integer(covCount$Freq*1000/totCount)/10)
covCount
```

```
##            Var1   Freq Percent
## 1         Aspen   6645     1.6
## 2 Cotton&Willow   1923     0.4
## 3    DouglasFir  12157     2.9
## 4     Krummholz  14357     3.5
## 5     Lodgepole 198311    48.7
## 6     Ponderosa  25028     6.1
## 7    Spruce&Fir 148288    36.4
```

View Test Set Coverage Percentages

Check test set coverage percentages.

```
covCount<-data.frame(table(forestTest$CovName))
totCount<-nrow(forestTest)
covCount <- mutate(covCount,Percent = as.integer(covCount$Freq*1000/totCount)/10)
covCount
```

```
##            Var1  Freq Percent
## 1         Aspen  2848     1.6
## 2 Cotton&Willow   824     0.4
## 3    DouglasFir  5210     2.9
## 4     Krummholz  6153     3.5
## 5     Lodgepole 84990    48.7
## 6     Ponderosa 10726     6.1
## 7    Spruce&Fir 63552    36.4
```

```
# knitr::knit_exit() # exit early

#glimpse(forestTrain)
#glimpse(forestTest)
#summary(forestTrain)
#summary(forestTest)
#table(forestTrain$Sed_mix)
#table(forestTrain$GeoName)
#table(forestTrain$LodgepolePine)
#table(forestTest$LodgepolePine)

# the above all work without error.

#table(forestTest$Rock_Land)
# Get the following error with above code:
#  Error in table(SpfFir_test$Rock_Land) : object 'SpfFir_test' not found
#    Calls: <Anonymous> ... withCallingHandlers -> withVisible -> eval -> eval -> table


#table(forestTrain$Rock_Land)
#table(forestTest$Rock_Land)
#table(forestTrain$Rubbly)
#table(forestTest$Rubbly)

#table(forestTrain$Sed_mix)
#table(forestTrain$Gateview)
#table(forestTrain$Rubbly)
#table(forestTest$Sed_mix)
#table(forestTest$Gateview)
#table(forestTest$Rubbly)

############# Start Start Start Start Start Start Start Start #################
```

Ponderosa Pine Logistic Regression

Logistic regression models are created and compared for the Ponderosa Pine coverage type. The outcome is based on the binary ‘PonderosaPine’ variable.

Ponderosa Pine Logistic Regression - All Variables

Create Ponderosa Pine Logistic Model - All Vars

Create the Ponderosa Pine logistic model for the Aggregated Soil data using all independent variables.

Ponderosa Pine All Aggregated Soil Types

The original project used aggregated Soil Types. Compute a logistic regression model using the aggregated soil types to see how the dis-aggregated / individuated variables compare.

```
# You can remove the levels of the factor variables using the option exclude:
#   lm(dependent ~ factor(independent1, exclude=c('b','d')) + independent2)
#   This way the factors b, d will not be included in the regression.

curTime=Sys.time()
print(paste("PonderosaPine aggregated Logistic Model Calculation started at",curTime))
```

```
## [1] "PonderosaPine aggregated Logistic Model Calculation started at 2018-08-12 20:54:41"
```

```
Ponder_Agg_LogMod =
  glm(PonderosaPine ~
        Elev +     # Elevation in meters of data cell
        Aspect +   # Direction in degrees slope faces
        Slope +    # Slope / steepness of hill in degrees (0 to 90)
        H2OHD +    # Horizontal distance in meters to nearest water
        H2OVD +    # Vertical distance in meters to nearest water
        RoadHD +   # Horizontal distance in meters to nearest road
        FirePtHD + # Horizontal distance in meters to nearest fire point
        Shade9AM + Shade12PM + Shade3PM + # Amount of shade at 9am, 12pm and 3pm
        # Wilderness areas:
          RWwild + NEwild + CMwild + CPwild +
        # Aggregated Soil type:
          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 ,
        data=forestTrain, family=binomial)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
# save model for later use
Ponder_Agg_All_LogMod = Ponder_Agg_LogMod
save("Ponder_Agg_All_LogMod", file="Ponder_Agg_All_LogMod.Rdata")

Ponder_Agg_All_aic<-as.integer(Ponder_Agg_LogMod$aic)
Ponder_Agg_All_aic
```

```
## [1] 61850
```

```
curTime=Sys.time()
print(paste("PonderosaPine aggregated Logistic Model Calculation completed at",curTime))
```

```
## [1] "PonderosaPine aggregated Logistic Model Calculation completed at 2018-08-12 20:57:45"
```

Check the coefficients for the Ponderosa Pine model using all aggregated data.

```
summary(Ponder_Agg_LogMod)
```

```
##
## Call:
## glm(formula = PonderosaPine ~ Elev + Aspect + Slope + H2OHD +
##     H2OVD + RoadHD + FirePtHD + Shade9AM + Shade12PM + Shade3PM +
##     RWwild + NEwild + CMwild + CPwild + 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, family = binomial, data = forestTrain)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.8069  -0.0113   0.0000   0.0000   3.7913
##
## Coefficients: (1 not defined because of singularities)
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.160e+09  3.122e+11  -0.004    0.997
## Elev        -5.560e-03  9.866e-05 -56.358  < 2e-16 ***
## Aspect       1.508e-03  1.278e-04  11.804  < 2e-16 ***
## Slope       -5.128e-03  3.839e-03  -1.336    0.182
## H2OHD        1.927e-03  9.413e-05  20.468  < 2e-16 ***
## H2OVD        1.878e-03  2.449e-04   7.667 1.76e-14 ***
## RoadHD      -1.274e-04  1.778e-05  -7.165 7.79e-13 ***
## FirePtHD    -3.018e-04  2.061e-05 -14.641  < 2e-16 ***
## Shade9AM    -4.099e-02  3.585e-03 -11.436  < 2e-16 ***
## Shade12PM    4.646e-02  3.044e-03  15.264  < 2e-16 ***
## Shade3PM    -4.001e-02  2.994e-03 -13.364  < 2e-16 ***
## RWwild      -1.452e+01  1.549e+02  -0.094    0.925
## NEwild      -1.249e+01  2.692e+02  -0.046    0.963
## CMwild       3.785e-01  3.816e-02   9.920  < 2e-16 ***
## CPwild              NA         NA      NA       NA
## ST01         1.160e+09  3.122e+11   0.004    0.997
## ST02         1.160e+09  3.122e+11   0.004    0.997
## ST03         1.160e+09  3.122e+11   0.004    0.997
## ST04         1.160e+09  3.122e+11   0.004    0.997
## ST05         1.160e+09  3.122e+11   0.004    0.997
## ST06         1.160e+09  3.122e+11   0.004    0.997
## ST07         1.160e+09  3.122e+11   0.004    0.997
## ST08         1.160e+09  3.122e+11   0.004    0.997
## ST09         1.160e+09  3.122e+11   0.004    0.997
## ST10         1.160e+09  3.122e+11   0.004    0.997
## ST11         1.160e+09  3.122e+11   0.004    0.997
## ST12         1.160e+09  3.122e+11   0.004    0.997
## ST13         1.160e+09  3.122e+11   0.004    0.997
## ST14         1.160e+09  3.122e+11   0.004    0.997
## ST15         1.160e+09  3.122e+11   0.004    0.997
## ST16         1.160e+09  3.122e+11   0.004    0.997
## ST17         1.160e+09  3.122e+11   0.004    0.997
## ST18         1.160e+09  3.122e+11   0.004    0.997
## ST19         1.160e+09  3.122e+11   0.004    0.997
## ST20         1.160e+09  3.122e+11   0.004    0.997
## ST21         1.160e+09  3.122e+11   0.004    0.997
## ST22         1.160e+09  3.122e+11   0.004    0.997
## ST23         1.160e+09  3.122e+11   0.004    0.997
## ST24         1.160e+09  3.122e+11   0.004    0.997
## ST25         1.160e+09  3.122e+11   0.004    0.997
## ST26         1.160e+09  3.122e+11   0.004    0.997
## ST27         1.160e+09  3.122e+11   0.004    0.997
## ST28         1.160e+09  3.122e+11   0.004    0.997
```

```
## ST29         1.160e+09  3.122e+11   0.004    0.997
## ST30         1.160e+09  3.122e+11   0.004    0.997
## ST31         1.160e+09  3.122e+11   0.004    0.997
## ST32         1.160e+09  3.122e+11   0.004    0.997
## ST33         1.160e+09  3.122e+11   0.004    0.997
## ST34         1.160e+09  3.122e+11   0.004    0.997
## ST35         1.160e+09  3.122e+11   0.004    0.997
## ST36         1.160e+09  3.122e+11   0.004    0.997
## ST37         1.160e+09  3.122e+11   0.004    0.997
## ST38         1.160e+09  3.122e+11   0.004    0.997
## ST39         1.160e+09  3.122e+11   0.004    0.997
## ST40         1.160e+09  3.122e+11   0.004    0.997
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 188044  on 406708  degrees of freedom
## Residual deviance:  61743  on 406655  degrees of freedom
## AIC: 61851
##
## Number of Fisher Scoring iterations: 21
```

WOW! The intercept is huge and listed as not significant. Wilderness area and several soil types are not significant and can be removed in the next iteration.

Ponderosa Pine All Individuated Soil Types

Create a logistic model using the Individuated variables that were derived from the Soil Types. The Soil Type was the intersection of climate zone, geology zone, soil families, and rock content. These variables are used instead of the Soil types.

```
curTime=Sys.time()
print(paste("PonderosaPine Individual Logistic Model Calculation started at",curTime))
```

```
## [1] "PonderosaPine Individual Logistic Model Calculation started at 2018-08-12 20:57:45"
```

```
Ponder_Ind_LogMod =
  glm(PonderosaPine ~
        Elev +     # Elevation in meters of cell
        Aspect +   # Direction in degrees slope faces
        Slope +    # Slope / steepness of hill in degrees (0 to 90)
        H2OHD +    # Horizontal distance in meters to nearest water
        H2OVD +    # Vertical distance in meters to nearest water
        RoadHD +   # Horizontal distance in meters to nearest road
        FirePtHD + # Horizontal distance in meters to nearest fire point
        Shade9AM + Shade12PM + Shade3PM + # Amount of shade at 9am, 12pm and 3pm
        # Wilderness areas:
          RWwild + NEwild + CMwild + CPwild +
        # Climate Zone:
        # ClimateName +
          Montane_low + Montane + SubAlpine + Alpine + Dry + Non_Dry +
        # Geology Zone:
        # GeoName +
          Alluvium + Glacial + Sed_mix + Ign_Meta +
```

```r
          # Soil Family:
            Aquolis_cmplx + Argiborolis_Pachic + Borohemists_cmplx + Bross +
            Bullwark + Bullwark_Cmplx + Catamount + Catamount_cmplx +
            Cathedral + Como + Cryaquepts_cmplx + Cryaquepts_Typic + Cryaquolls +
            Cryaquolls_cmplx + Cryaquolls_Typic + Cryaquolls_Typic_cmplx +
            Cryoborolis_cmplx + Cryorthents + Cryorthents_cmplx + Cryumbrepts +
            Cryumbrepts_cmplx + Gateview + Gothic + Granile + Haploborolis +
            Legault + Legault_cmplx + Leighcan + Leighcan_cmplx + Leighcan_warm +
            Moran + Ratake + Ratake_cmplx + Rogert + Supervisor_Limber_cmplx +
            Troutville + Unspecified + Vanet + Wetmore +
          # Soil Rock composition:
            Bouldery_ext + Rock_Land + Rock_Land_cmplx + Rock_Outcrop +
            Rock_Outcrop_cmplx + Rubbly + Stony + Stony_extreme + Stony_very +
            Till_Substratum ,
          data=forestTrain, family=binomial)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
# save model for later use
Ponder_Ind_All_LogMod = Ponder_Ind_LogMod
save("Ponder_Ind_All_LogMod", file="Ponder_Ind_All_LogMod.Rdata")

#table(forestTrain$GeoName)
#table(forestTrain$Sed_mix)
#table(forestTrain$Gateview)
# above: Error in table(SpfFir_test$Gateview) : object 'SpfFir_train' not found <-------

Ponder_Ind_All_aic<-as.integer(Ponder_Ind_LogMod$aic)
Ponder_Ind_All_aic
```

```
## [1] 61856
```

```r
summary(Ponder_Ind_LogMod)
```

```
##
## Call:
## glm(formula = PonderosaPine ~ Elev + Aspect + Slope + H2OHD +
##     H2OVD + RoadHD + FirePtHD + Shade9AM + Shade12PM + Shade3PM +
##     RWwild + NEwild + CMwild + CPwild + Montane_low + Montane +
##     SubAlpine + Alpine + Dry + Non_Dry + Alluvium + Glacial +
##     Sed_mix + Ign_Meta + Aquolis_cmplx + Argiborolis_Pachic +
##     Borohemists_cmplx + Bross + Bullwark + Bullwark_Cmplx + Catamount +
##     Catamount_cmplx + Cathedral + Como + Cryaquepts_cmplx + Cryaquepts_Typic +
##     Cryaquolls + Cryaquolls_cmplx + Cryaquolls_Typic + Cryaquolls_Typic_cmplx +
##     Cryoborolis_cmplx + Cryorthents + Cryorthents_cmplx + Cryumbrepts +
##     Cryumbrepts_cmplx + Gateview + Gothic + Granile + Haploborolis +
##     Legault + Legault_cmplx + Leighcan + Leighcan_cmplx + Leighcan_warm +
##     Moran + Ratake + Ratake_cmplx + Rogert + Supervisor_Limber_cmplx +
##     Troutville + Unspecified + Vanet + Wetmore + Bouldery_ext +
##     Rock_Land + Rock_Land_cmplx + Rock_Outcrop + Rock_Outcrop_cmplx +
##     Rubbly + Stony + Stony_extreme + Stony_very + Till_Substratum,
##     family = binomial, data = forestTrain)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
```

```
## -2.8068  -0.0113   0.0000   0.0000   3.7911
##
## Coefficients: (17 not defined because of singularities)
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)             1.393e+09  7.098e+11   0.002    0.998
## Elev                   -5.560e-03  1.016e-04 -54.732  < 2e-16 ***
## Aspect                  1.509e-03  1.385e-04  10.894  < 2e-16 ***
## Slope                  -5.128e-03  3.981e-03  -1.288    0.198
## H2OHD                   1.927e-03  9.707e-05  19.847  < 2e-16 ***
## H2OVD                   1.878e-03  2.480e-04   7.574 3.63e-14 ***
## RoadHD                 -1.274e-04  1.798e-05  -7.084 1.40e-12 ***
## FirePtHD               -3.017e-04  2.082e-05 -14.492  < 2e-16 ***
## Shade9AM               -4.100e-02  3.687e-03 -11.120  < 2e-16 ***
## Shade12PM               4.647e-02  3.237e-03  14.356  < 2e-16 ***
## Shade3PM               -4.002e-02  3.152e-03 -12.697  < 2e-16 ***
## RWwild                 -1.453e+01  1.550e+02  -0.094    0.925
## NEwild                 -1.248e+01  2.671e+02  -0.047    0.963
## CMwild                  3.785e-01  3.878e-02   9.760  < 2e-16 ***
## CPwild                         NA         NA      NA       NA
## Montane_low            -2.508e+09  1.105e+12  -0.002    0.998
## Montane                 3.548e+09  5.683e+11   0.006    0.995
## SubAlpine              -1.393e+09  7.098e+11  -0.002    0.998
## Alpine                 -1.393e+09  7.098e+11  -0.002    0.998
## Dry                     4.009e+10  1.558e+13   0.003    0.998
## Non_Dry                 1.115e+09  5.444e+11   0.002    0.998
## Alluvium               -4.663e+09  8.035e+11  -0.006    0.995
## Glacial                -7.224e+09  7.509e+12  -0.001    0.999
## Sed_mix                -4.503e+10  1.627e+13  -0.003    0.998
## Ign_Meta                       NA         NA      NA       NA
## Aquolis_cmplx          -4.148e+10  1.609e+13  -0.003    0.998
## Argiborolis_Pachic             NA         NA      NA       NA
## Borohemists_cmplx      -3.224e+00  2.174e+03  -0.001    0.999
## Bross                  -7.726e+00  5.472e+03  -0.001    0.999
## Bullwark               -6.056e+09  1.278e+12  -0.005    0.996
## Bullwark_Cmplx         -6.056e+09  1.278e+12  -0.005    0.996
## Catamount               1.644e+01  3.038e+03   0.005    0.996
## Catamount_cmplx        -4.107e-01  4.941e+02  -0.001    0.999
## Cathedral               4.116e-01  9.010e-02   4.568 4.93e-06 ***
## Como                    1.012e+01  7.975e+02   0.013    0.990
## Cryaquepts_cmplx       -6.083e+00  2.159e+03  -0.003    0.998
## Cryaquepts_Typic       -2.561e+09  7.416e+12   0.000    1.000
## Cryaquolls             -1.792e+00  1.565e+03  -0.001    0.999
## Cryaquolls_cmplx       -1.944e+00  1.565e+03  -0.001    0.999
## Cryaquolls_Typic        4.663e+09  8.035e+11   0.006    0.995
## Cryaquolls_Typic_cmplx  7.224e+09  7.509e+12   0.001    0.999
## Cryoborolis_cmplx              NA         NA      NA       NA
## Cryorthents           -2.155e+00  3.403e+03  -0.001    0.999
## Cryorthents_cmplx     -5.399e+00  3.447e+03  -0.002    0.999
## Cryumbrepts                    NA         NA      NA       NA
## Cryumbrepts_cmplx              NA         NA      NA       NA
## Gateview                       NA         NA      NA       NA
## Gothic                  7.398e-02  7.113e+03   0.000    1.000
## Granile                -5.007e+00  1.331e+03  -0.004    0.997
## Haploborolis            5.281e-01  8.636e-02   6.115 9.67e-10 ***
```

```
## Legault                   -6.056e+09  1.278e+12  -0.005    0.996
## Legault_cmplx                    NA         NA      NA       NA
## Leighcan                  -4.556e+00  6.770e+02  -0.007    0.995
## Leighcan_cmplx            -4.920e+00  3.133e+03  -0.002    0.999
## Leighcan_warm             -6.657e-01  3.374e+03   0.000    1.000
## Moran                            NA         NA      NA       NA
## Ratake                     2.266e+00  8.352e-02  27.129  < 2e-16 ***
## Ratake_cmplx              -1.352e+00  3.059e+03   0.000    1.000
## Rogert                    -4.663e+09  8.035e+11  -0.006    0.995
## Supervisor_Limber_cmplx          NA         NA      NA       NA
## Troutville                 1.168e+09  7.436e+12   0.000    1.000
## Unspecified               -4.148e+10  1.609e+13  -0.003    0.998
## Vanet                            NA         NA      NA       NA
## Wetmore                    1.546e+00  8.346e-02  18.527  < 2e-16 ***
## Bouldery_ext               7.224e+09  7.509e+12   0.001    0.999
## Rock_Land                 -7.676e-01  3.491e+02  -0.002    0.998
## Rock_Land_cmplx           -3.776e+00  3.059e+03  -0.001    0.999
## Rock_Outcrop                     NA         NA      NA       NA
## Rock_Outcrop_cmplx        -3.557e+00  3.059e+03  -0.001    0.999
## Rubbly                           NA         NA      NA       NA
## Stony                            NA         NA      NA       NA
## Stony_extreme                    NA         NA      NA       NA
## Stony_very                       NA         NA      NA       NA
## Till_Substratum                  NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 188044  on 406708  degrees of freedom
## Residual deviance:  61743  on 406652  degrees of freedom
## AIC: 61857
##
## Number of Fisher Scoring iterations: 21
```

```
  curTime=Sys.time()
  print(paste("PonderosaPine Individual Logistic Model Calculation completed at",curTime))
```

```
## [1] "PonderosaPine Individual Logistic Model Calculation completed at 2018-08-12 21:02:34"
```

```
  #table(forestTest$Rock_Land)
  # Get the following error with above code:
  #  Error in table(SpfFir_test$Rock_Land) : object 'SpfFir_test' not found
  #    Calls: <Anonymous> ... withCallingHandlers -> withVisible -> eval -> eval -> table
```

Predict Ponderosa Pine Logistic Model Probabilities - All Aggregated Vars

Ponderosa Pine Probabilities - All Aggregated Data

Predict the probability of Ponderosa Pine for aggregated Data - all variables.

```
# Predict Ponderosa Pine Agg Data - all variables

  Ponder_Agg_Train_predict= predict(Ponder_Agg_LogMod, type="response")
  Ponder_Agg_Train_Logit= predict(Ponder_Agg_LogMod)
```

```r
summary(Ponder_Agg_Train_predict)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000000 0.0000000 0.0000000 0.0615380 0.0005567 0.9995617
```

```r
str(Ponder_Agg_Train_predict)
```

```
##  Named num [1:406709] 3.02e-10 3.14e-10 5.55e-11 2.73e-10 5.34e-10 ...
##  - attr(*, "names")= chr [1:406709] "1" "2" "3" "4" ...
```

```r
  #plot(table(Ponder_Agg_Train_predict))
  #plot(table(Ponder_Agg_Train_Logit))
  dens<-data.frame(table(Ponder_Agg_Train_predict))
# str(dens)

  Ponder_Agg_Test_predict= predict(Ponder_Agg_LogMod, type="response",newdata=forestTest)
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =
## ifelse(type == : prediction from a rank-deficient fit may be misleading
```

```r
summary(Ponder_Agg_Test_predict)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000000 0.0000000 0.0000000 0.0614225 0.0005627 0.9998034
```

```r
 str(Ponder_Agg_Test_predict)
```

```
##  Named num [1:174303] 8.92e-11 2.77e-10 6.24e-11 9.89e-11 1.22e-08 ...
##  - attr(*, "names")= chr [1:174303] "1" "2" "3" "4" ...
```

Ponderosa Pine Probabilities - All Individuated Data

Predict the probability of Ponderosa Pine for Individual Data - all variables.

```r
Ponder_Ind_Train_predict= predict(Ponder_Ind_LogMod, type="response")
summary(Ponder_Ind_Train_predict)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000000 0.0000000 0.0000000 0.0615378 0.0005569 0.9995621
```

```r
Ponder_Ind_Test_predict= predict(Ponder_Ind_LogMod, type="response",newdata=forestTest)
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =
## ifelse(type == : prediction from a rank-deficient fit may be misleading
```

```r
summary(Ponder_Ind_Test_predict)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000000 0.0000000 0.0000000 0.0614223 0.0005628 0.9998036
```

Ponderosa Pine Receiver Operating Characteristic (ROC) - All Vars

Ponderosa Pine Receiver ROC - All Aggregated Data

Next, look at the True Positive and False Positive rates based on threshold value for the aggregated data.

```r
if (calcROC) {
  curTime=Sys.time()
  print(paste("ROC graph 1 started at",curTime))
```

```r
    ROCpred_Ponder_Agg = prediction(Ponder_Agg_Train_predict, forestTrain$PonderosaPine)
    summary(ROCpred_Ponder_Agg)
    ROCperf_Ponder_Agg = performance(ROCpred_Ponder_Agg, "tpr", "fpr")
    summary(ROCperf_Ponder_Agg)

    Ponder_Agg_All_ROC_AUC = as.numeric(performance(ROCpred_Ponder_Agg, "auc")@y.values)
    Ponder_Agg_All_ROC_AUC=as.integer(as.numeric(Ponder_Agg_All_ROC_AUC)*1000)/10
    print(paste("Ponder_Agg_All_ROC_AUC=",Ponder_Agg_All_ROC_AUC))

    jpeg(filename="Fig-ROCR_perf_Ponder_Agg.jpg")
    plot(ROCperf_Ponder_Agg, colorize=TRUE, print.cutoffs.at=seq(0,1,0.1), text.adj=c(-0.2,1.7))
    dev.off()
  } else {
    Ponder_Agg_All_ROC_AUC = 84.2
  }
```

```
## [1] "ROC graph 1 started at 2018-08-12 21:02:40"
## [1] "Ponder_Agg_All_ROC_AUC= 98.3"
```

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

Ponderosa Pine Receiver ROC - All Individuated Data

The Response Operating Curve for the individuated data is shown below.

```r
  if (calcROC) {
    curTime=Sys.time()
    print(paste("ROCR graph 2 started at",curTime))

    ROCpred_Ponder_Ind = prediction(Ponder_Ind_Train_predict, forestTrain$PonderosaPine)
    summary(ROCpred_Ponder_Ind)
    ROCperf_Ponder_Ind = performance(ROCpred_Ponder_Ind, "tpr", "fpr")
    summary(ROCperf_Ponder_Ind)

    Ponder_Ind_All_ROC_AUC = as.numeric(performance(ROCpred_Ponder_Ind, "auc")@y.values)
    Ponder_Ind_All_ROC_AUC=as.integer(as.numeric(Ponder_Ind_All_ROC_AUC)*1000)/10
    print(paste("Ponder_Ind_All_ROC_AUC=",Ponder_Ind_All_ROC_AUC))

    jpeg(filename="Fig-ROCR_perf_Ponder_Ind.jpg")
    plot(ROCperf_Ponder_Ind, colorize=TRUE, print.cutoffs.at=seq(0,1,0.1), text.adj=c(-0.2,1.7))
    dev.off()
  } else {
    Ponder_Ind_All_ROC_AUC = 84.2
  }
```

```
## [1] "ROCR graph 2 started at 2018-08-12 21:05:56"
## [1] "Ponder_Ind_All_ROC_AUC= 98.3"
```

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

The threshold graphs are essentially identical. This is making me think that there is not much difference between the two models. The AIC score for the Soil Type model is AIC: 351676 and for the individuated variables is: AIC: 351839. The Soil type model AIC score is 0.046% better than the individuated model.

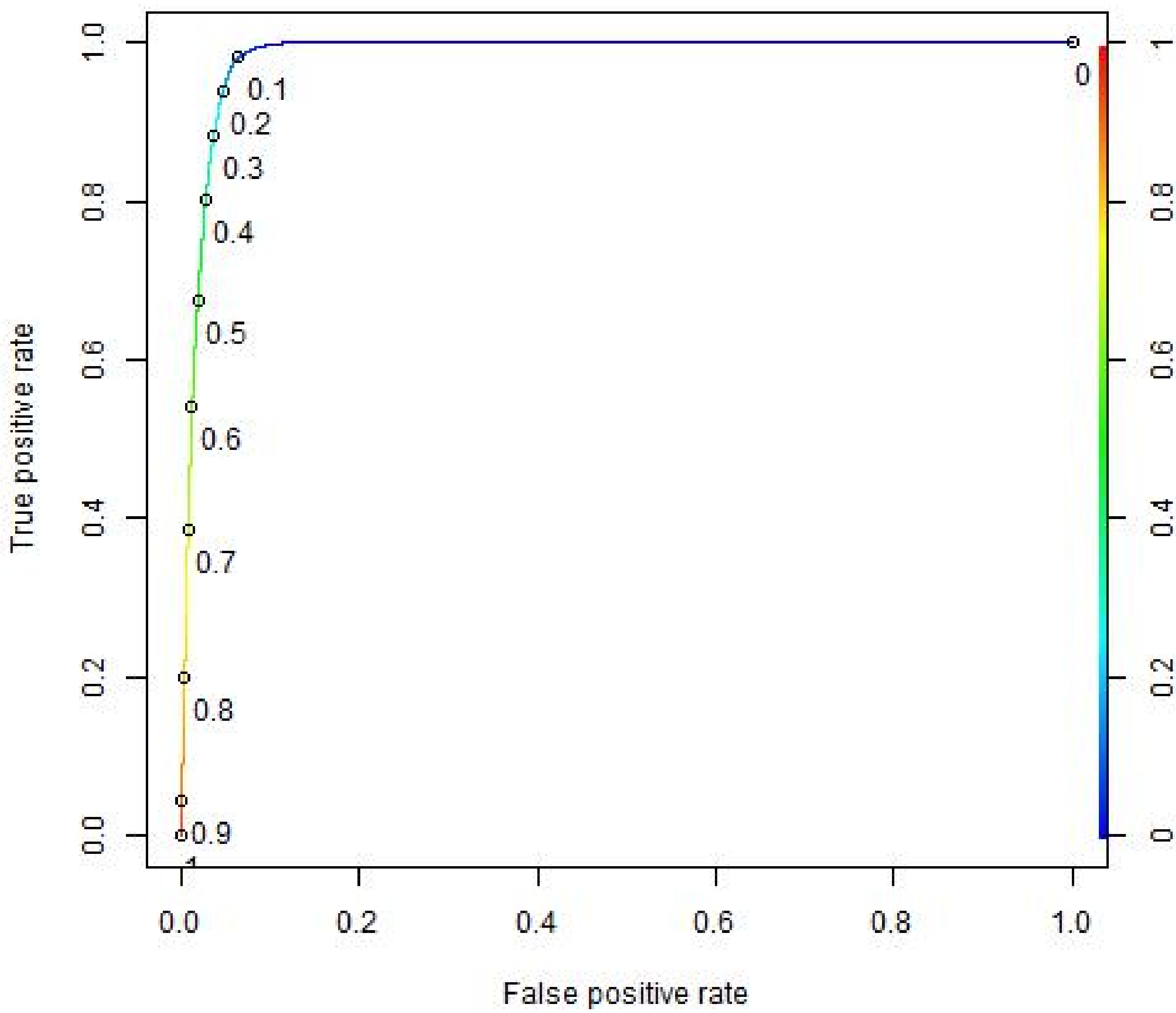

Figure 1: Ponderosa Pine ROC for All Aggregated Data

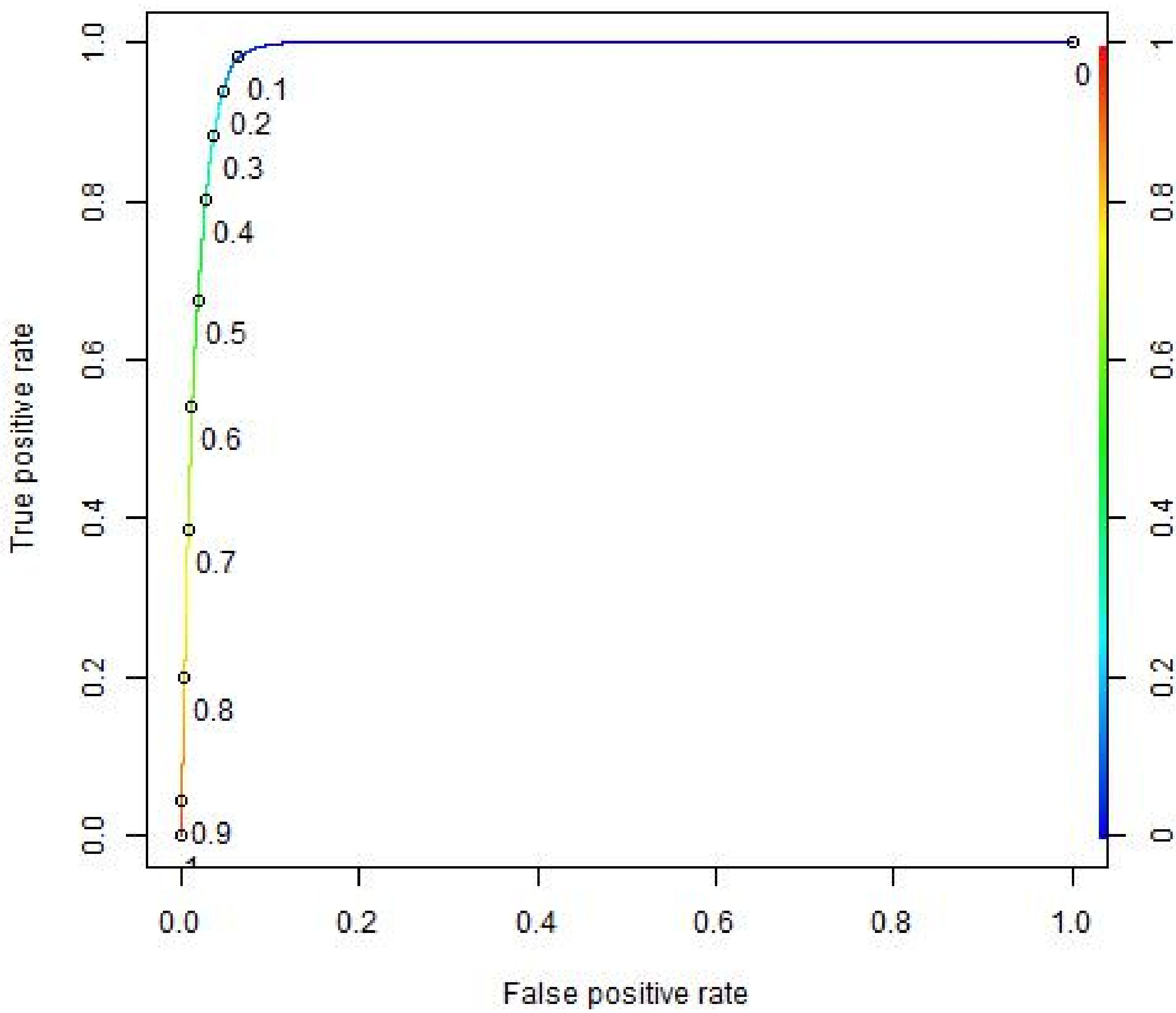

Figure 2: Ponderosa Pine ROC for All Individuated Data

```
curTime=Sys.time()
print(paste("ROCR graph 2 completed at",curTime))
```

```
## [1] "ROCR graph 2 completed at 2018-08-12 21:08:50"
```

Calculate Accuracy of Ponderosa Pine Logisitic Models - All Vars

Calculate Ponderosa Pine Aggregated Data Logisitic Model Accuracy - All Vars

Find best threshold for Ponderosa Pine using all aggregated data.

```
result = calcLogisticModelAccuracy (forestTrain$PonderosaPine, Ponder_Agg_Train_predict,
                      0.0, 1, 10, "Ponderosa", "Other", 1,1)
```

```
## [1] "Searching for threshold producing best Sensitivity_Specificity"
## [1] "start= 0 end= 1 inc= 0.1"
## [1] "Thresh=0, Accuracy=6.1%, BaseAcc(Other)=93.8%, Sens=100%, Spec=0%, Sens^2+Spec^2=-2"
## [1] "Thresh=0.1, Accuracy=93.8%, BaseAcc(Other)=93.8%, Sens=98.2%, Spec=93.5%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.2, Accuracy=95.2%, BaseAcc(Other)=93.8%, Sens=93.9%, Spec=95.3%, Sens^2+Spec^2=1.792"
## [1] "Thresh=0.3, Accuracy=95.8%, BaseAcc(Other)=93.8%, Sens=88.3%, Spec=96.3%, Sens^2+Spec^2=1.708"
## [1] "Thresh=0.4, Accuracy=96.2%, BaseAcc(Other)=93.8%, Sens=80.1%, Spec=97.2%, Sens^2+Spec^2=1.589"
## [1] "Thresh=0.5, Accuracy=96.2%, BaseAcc(Other)=93.8%, Sens=67.6%, Spec=98.1%, Sens^2+Spec^2=1.42"
## [1] "Thresh=0.6, Accuracy=96%, BaseAcc(Other)=93.8%, Sens=54%, Spec=98.8%, Sens^2+Spec^2=1.268"
## [1] "Thresh=0.7, Accuracy=95.5%, BaseAcc(Other)=93.8%, Sens=38.5%, Spec=99.2%, Sens^2+Spec^2=1.133"
## [1] "Thresh=0.8, Accuracy=94.7%, BaseAcc(Other)=93.8%, Sens=20%, Spec=99.6%, Sens^2+Spec^2=1.032"
## [1] "Thresh=0.9, Accuracy=94%, BaseAcc(Other)=93.8%, Sens=4.3%, Spec=99.9%, Sens^2+Spec^2=1"
## [1] "Thresh=1, Accuracy=93.8%, BaseAcc(Other)=93.8%, Sens=0%, Spec=100%, Sens^2+Spec^2=-2"
## [1] "Best Sensitivity_Specificity threshold= 0.1 inc= 0.1"
## [1] "========================================"
## [1] "start= 0 end= 0.2 inc= 0.01"
## [1] "Thresh=0, Accuracy=6.1%, BaseAcc(Other)=93.8%, Sens=100%, Spec=0%, Sens^2+Spec^2=-2"
## [1] "Thresh=0.01, Accuracy=91.2%, BaseAcc(Other)=93.8%, Sens=99.6%, Spec=90.6%, Sens^2+Spec^2=1.815"
## [1] "Thresh=0.02, Accuracy=91.5%, BaseAcc(Other)=93.8%, Sens=99.5%, Spec=90.9%, Sens^2+Spec^2=1.819"
## [1] "Thresh=0.03, Accuracy=91.6%, BaseAcc(Other)=93.8%, Sens=99.5%, Spec=91.1%, Sens^2+Spec^2=1.821"
## [1] "Thresh=0.04, Accuracy=91.8%, BaseAcc(Other)=93.8%, Sens=99.4%, Spec=91.3%, Sens^2+Spec^2=1.824"
## [1] "Thresh=0.05, Accuracy=92.1%, BaseAcc(Other)=93.8%, Sens=99.3%, Spec=91.7%, Sens^2+Spec^2=1.827"
## [1] "Thresh=0.06, Accuracy=92.5%, BaseAcc(Other)=93.8%, Sens=99%, Spec=92%, Sens^2+Spec^2=1.829"
## [1] "Thresh=0.07, Accuracy=92.8%, BaseAcc(Other)=93.8%, Sens=98.9%, Spec=92.5%, Sens^2+Spec^2=1.834"
## [1] "Thresh=0.08, Accuracy=93.2%, BaseAcc(Other)=93.8%, Sens=98.7%, Spec=92.8%, Sens^2+Spec^2=1.837"
## [1] "Thresh=0.09, Accuracy=93.5%, BaseAcc(Other)=93.8%, Sens=98.4%, Spec=93.2%, Sens^2+Spec^2=1.839"
## [1] "Thresh=0.1, Accuracy=93.8%, BaseAcc(Other)=93.8%, Sens=98.2%, Spec=93.5%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.11, Accuracy=94.1%, BaseAcc(Other)=93.8%, Sens=97.8%, Spec=93.8%, Sens^2+Spec^2=1.839"
## [1] "Thresh=0.12, Accuracy=94.3%, BaseAcc(Other)=93.8%, Sens=97.4%, Spec=94.1%, Sens^2+Spec^2=1.836"
## [1] "Thresh=0.13, Accuracy=94.5%, BaseAcc(Other)=93.8%, Sens=96.9%, Spec=94.3%, Sens^2+Spec^2=1.831"
## [1] "Thresh=0.14, Accuracy=94.6%, BaseAcc(Other)=93.8%, Sens=96.5%, Spec=94.5%, Sens^2+Spec^2=1.826"
## [1] "Thresh=0.15, Accuracy=94.8%, BaseAcc(Other)=93.8%, Sens=96%, Spec=94.7%, Sens^2+Spec^2=1.82"
## [1] "Thresh=0.16, Accuracy=94.9%, BaseAcc(Other)=93.8%, Sens=95.5%, Spec=94.8%, Sens^2+Spec^2=1.813"
## [1] "Thresh=0.17, Accuracy=95%, BaseAcc(Other)=93.8%, Sens=95%, Spec=95%, Sens^2+Spec^2=1.806"
## [1] "Thresh=0.18, Accuracy=95.1%, BaseAcc(Other)=93.8%, Sens=94.7%, Spec=95.1%, Sens^2+Spec^2=1.802"
## [1] "Thresh=0.19, Accuracy=95.2%, BaseAcc(Other)=93.8%, Sens=94.3%, Spec=95.2%, Sens^2+Spec^2=1.797"
## [1] "Best Sensitivity_Specificity threshold= 0.1 inc= 0.01"
## [1] "========================================"
## [1] "start= 0.09 end= 0.11 inc= 0.001"
## [1] "Thresh=0.09, Accuracy=93.5%, BaseAcc(Other)=93.8%, Sens=98.4%, Spec=93.2%, Sens^2+Spec^2=1.839"
```

```
## [1] "Thresh=0.091, Accuracy=93.5%, BaseAcc(Other)=93.8%, Sens=98.4%, Spec=93.2%, Sens^2+Spec^2=1.839
## [1] "Thresh=0.092, Accuracy=93.6%, BaseAcc(Other)=93.8%, Sens=98.4%, Spec=93.3%, Sens^2+Spec^2=1.839
## [1] "Thresh=0.093, Accuracy=93.6%, BaseAcc(Other)=93.8%, Sens=98.4%, Spec=93.3%, Sens^2+Spec^2=1.839
## [1] "Thresh=0.094, Accuracy=93.6%, BaseAcc(Other)=93.8%, Sens=98.3%, Spec=93.3%, Sens^2+Spec^2=1.839
## [1] "Thresh=0.095, Accuracy=93.7%, BaseAcc(Other)=93.8%, Sens=98.3%, Spec=93.4%, Sens^2+Spec^2=1.839
## [1] "Thresh=0.096, Accuracy=93.7%, BaseAcc(Other)=93.8%, Sens=98.3%, Spec=93.4%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.097, Accuracy=93.7%, BaseAcc(Other)=93.8%, Sens=98.3%, Spec=93.4%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.098, Accuracy=93.8%, BaseAcc(Other)=93.8%, Sens=98.2%, Spec=93.5%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.099, Accuracy=93.8%, BaseAcc(Other)=93.8%, Sens=98.2%, Spec=93.5%, Sens^2+Spec^2=1.841
## [1] "Thresh=0.1, Accuracy=93.8%, BaseAcc(Other)=93.8%, Sens=98.2%, Spec=93.5%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.101, Accuracy=93.9%, BaseAcc(Other)=93.8%, Sens=98.1%, Spec=93.6%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.102, Accuracy=93.9%, BaseAcc(Other)=93.8%, Sens=98.1%, Spec=93.6%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.103, Accuracy=93.9%, BaseAcc(Other)=93.8%, Sens=98.1%, Spec=93.6%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.104, Accuracy=93.9%, BaseAcc(Other)=93.8%, Sens=98.1%, Spec=93.7%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.105, Accuracy=94%, BaseAcc(Other)=93.8%, Sens=98%, Spec=93.7%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.106, Accuracy=94%, BaseAcc(Other)=93.8%, Sens=98%, Spec=93.7%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.107, Accuracy=94%, BaseAcc(Other)=93.8%, Sens=98%, Spec=93.7%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.108, Accuracy=94%, BaseAcc(Other)=93.8%, Sens=97.9%, Spec=93.8%, Sens^2+Spec^2=1.839"
## [1] "Thresh=0.109, Accuracy=94.1%, BaseAcc(Other)=93.8%, Sens=97.9%, Spec=93.8%, Sens^2+Spec^2=1.839
## [1] "========================================"
## [1] "Best Threshold=0.099"
## [1] "Best Sensitivity_Specificity=1.84101718152265"
```

```
curThresh = as.numeric(result[bestThreshIndex])
Ponder_Agg_All_threshold = curThresh
```

The accuracy for the best threshold on the training set for Ponderosa Pine using all aggregated data is shown below.

```
result = calcLogisticModelAccuracy (forestTrain$PonderosaPine, Ponder_Agg_Train_predict,
                       curThresh, curThresh, 1, "Ponderosa", "Other", 3)
```

```
## [1] "Model Performance for threshold= 0.099"
## [1] "predicted performance="
##                        Predicted
## Actual                  FALSE=Predict:Other TRUE=Predict:Ponderosa
##   0=Actual:Other          357111 (TN)         24570 (FP)
##   1=Actual:Ponderosa      434 (FN)            24594 (TP)
## [1] "Sensitivity= 0.982659421447978 (True positive rate of Ponderosa = TP/(TP+FN) = 24594 /( 24594 +
## [1] "Specificity= 0.935626871654602 (True negative rate of Other = TN/(TN+FP) = 357111 /( 357111 + 2
## [1] "Sens^2+Spec^2=1.841"
## [1] "Baseline (Other) Accuracy=0.938462"
## [1] "Logistic Accuracy=0.938521"
```

The accuracy for the best threshold on the testing set for Ponderosa Pine using all aggregated data is shown below.

```
result = calcLogisticModelAccuracy (forestTest$PonderosaPine, Ponder_Agg_Test_predict,
                       curThresh, curThresh, 1, "Ponderosa", "Other", 3,
                       saveFile=saveFileName, desc="Ponderosa All Aggregate Vars",
                       AIC=Ponder_Agg_All_aic, AUC=Ponder_Agg_All_ROC_AUC)
```

```
## [1] "Model Performance for threshold= 0.099"
## [1] "predicted performance="
##                        Predicted
## Actual                  FALSE=Predict:Other TRUE=Predict:Ponderosa
##   0=Actual:Other          153003 (TN)         10574 (FP)
```

```
##   1=Actual:Ponderosa     202 (FN)           10524 (TP)
## [1] "Sensitivity= 0.981167257132202 (True positive rate of Ponderosa = TP/(TP+FN) = 10524 /( 10524 +
## [1] "Specificity= 0.935357660306767 (True negative rate of Other = TN/(TN+FP) = 153003 /( 153003 + 10
## [1] "Sens^2+Spec^2=1.837"
## [1] "Baseline (Other) Accuracy=0.938463"
## [1] "Logistic Accuracy=0.938176"
```

```r
# retVal = c(modelPerformance, sensitivity,specificity) # TN, FN, FP, TP, sens, spec
# c(funcStat,accuracy,baseline,retVal)
list[RC, Ponder_Agg_All_model_acc, Ponder_Agg_All_baseline_acc,
    TN, FN, FP, TP, Ponder_Agg_All_sens, Ponder_Agg_All_spec] <- result
if (RC != "OK") {
  print(paste("Error - terminating:",RC))
  knitr:knit_exit()
}
Ponder_Agg_All_model_acc = as.integer(as.numeric(Ponder_Agg_All_model_acc)*1000)/10
Ponder_Agg_All_baseline_acc = as.integer(as.numeric(Ponder_Agg_All_baseline_acc)*1000)/10
Ponder_Agg_All_sens = as.integer(as.numeric(Ponder_Agg_All_sens)*1000)/10
Ponder_Agg_All_spec = as.integer(as.numeric(Ponder_Agg_All_spec)*1000)/10
```

Calculate Ponderosa Pine Individuated Data Logisitic Model Accuracy - All Vars

Find best threshold for Ponderosa Pine using all individuated data.

```r
result = calcLogisticModelAccuracy (forestTrain$PonderosaPine, Ponder_Ind_Train_predict,
                        0.0, 1, 10, "Ponderosa", "Other", 1,1)
```

```
## [1] "Searching for threshold producing best Sensitivity_Specificity"
## [1] "start= 0 end= 1 inc= 0.1"
## [1] "Thresh=0, Accuracy=6.1%, BaseAcc(Other)=93.8%, Sens=100%, Spec=0%, Sens^2+Spec^2=-2"
## [1] "Thresh=0.1, Accuracy=93.8%, BaseAcc(Other)=93.8%, Sens=98.2%, Spec=93.5%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.2, Accuracy=95.2%, BaseAcc(Other)=93.8%, Sens=93.9%, Spec=95.3%, Sens^2+Spec^2=1.792"
## [1] "Thresh=0.3, Accuracy=95.8%, BaseAcc(Other)=93.8%, Sens=88.3%, Spec=96.3%, Sens^2+Spec^2=1.708"
## [1] "Thresh=0.4, Accuracy=96.2%, BaseAcc(Other)=93.8%, Sens=80.1%, Spec=97.2%, Sens^2+Spec^2=1.589"
## [1] "Thresh=0.5, Accuracy=96.2%, BaseAcc(Other)=93.8%, Sens=67.6%, Spec=98.1%, Sens^2+Spec^2=1.42"
## [1] "Thresh=0.6, Accuracy=96%, BaseAcc(Other)=93.8%, Sens=54%, Spec=98.8%, Sens^2+Spec^2=1.268"
## [1] "Thresh=0.7, Accuracy=95.5%, BaseAcc(Other)=93.8%, Sens=38.5%, Spec=99.2%, Sens^2+Spec^2=1.133"
## [1] "Thresh=0.8, Accuracy=94.7%, BaseAcc(Other)=93.8%, Sens=20%, Spec=99.6%, Sens^2+Spec^2=1.032"
## [1] "Thresh=0.9, Accuracy=94%, BaseAcc(Other)=93.8%, Sens=4.3%, Spec=99.9%, Sens^2+Spec^2=1"
## [1] "Thresh=1, Accuracy=93.8%, BaseAcc(Other)=93.8%, Sens=0%, Spec=100%, Sens^2+Spec^2=-2"
## [1] "Best Sensitivity_Specificity threshold= 0.1 inc= 0.1"
## [1] "========================================"
## [1] "start= 0 end= 0.2 inc= 0.01"
## [1] "Thresh=0, Accuracy=6.1%, BaseAcc(Other)=93.8%, Sens=100%, Spec=0%, Sens^2+Spec^2=-2"
## [1] "Thresh=0.01, Accuracy=91.2%, BaseAcc(Other)=93.8%, Sens=99.6%, Spec=90.6%, Sens^2+Spec^2=1.815"
## [1] "Thresh=0.02, Accuracy=91.5%, BaseAcc(Other)=93.8%, Sens=99.5%, Spec=90.9%, Sens^2+Spec^2=1.819"
## [1] "Thresh=0.03, Accuracy=91.6%, BaseAcc(Other)=93.8%, Sens=99.5%, Spec=91.1%, Sens^2+Spec^2=1.821"
## [1] "Thresh=0.04, Accuracy=91.8%, BaseAcc(Other)=93.8%, Sens=99.4%, Spec=91.3%, Sens^2+Spec^2=1.824"
## [1] "Thresh=0.05, Accuracy=92.1%, BaseAcc(Other)=93.8%, Sens=99.3%, Spec=91.7%, Sens^2+Spec^2=1.827"
## [1] "Thresh=0.06, Accuracy=92.5%, BaseAcc(Other)=93.8%, Sens=99%, Spec=92%, Sens^2+Spec^2=1.829"
## [1] "Thresh=0.07, Accuracy=92.8%, BaseAcc(Other)=93.8%, Sens=98.9%, Spec=92.5%, Sens^2+Spec^2=1.834"
## [1] "Thresh=0.08, Accuracy=93.2%, BaseAcc(Other)=93.8%, Sens=98.7%, Spec=92.8%, Sens^2+Spec^2=1.837"
## [1] "Thresh=0.09, Accuracy=93.5%, BaseAcc(Other)=93.8%, Sens=98.4%, Spec=93.2%, Sens^2+Spec^2=1.839"
## [1] "Thresh=0.1, Accuracy=93.8%, BaseAcc(Other)=93.8%, Sens=98.2%, Spec=93.5%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.11, Accuracy=94.1%, BaseAcc(Other)=93.8%, Sens=97.8%, Spec=93.8%, Sens^2+Spec^2=1.839"
```

```
## [1] "Thresh=0.12, Accuracy=94.3%, BaseAcc(Other)=93.8%, Sens=97.4%, Spec=94.1%, Sens^2+Spec^2=1.836"
## [1] "Thresh=0.13, Accuracy=94.5%, BaseAcc(Other)=93.8%, Sens=96.9%, Spec=94.3%, Sens^2+Spec^2=1.831"
## [1] "Thresh=0.14, Accuracy=94.6%, BaseAcc(Other)=93.8%, Sens=96.5%, Spec=94.5%, Sens^2+Spec^2=1.826"
## [1] "Thresh=0.15, Accuracy=94.8%, BaseAcc(Other)=93.8%, Sens=96%, Spec=94.7%, Sens^2+Spec^2=1.82"
## [1] "Thresh=0.16, Accuracy=94.9%, BaseAcc(Other)=93.8%, Sens=95.5%, Spec=94.8%, Sens^2+Spec^2=1.813"
## [1] "Thresh=0.17, Accuracy=95%, BaseAcc(Other)=93.8%, Sens=95%, Spec=95%, Sens^2+Spec^2=1.806"
## [1] "Thresh=0.18, Accuracy=95.1%, BaseAcc(Other)=93.8%, Sens=94.7%, Spec=95.1%, Sens^2+Spec^2=1.802"
## [1] "Thresh=0.19, Accuracy=95.2%, BaseAcc(Other)=93.8%, Sens=94.3%, Spec=95.2%, Sens^2+Spec^2=1.797"
## [1] "Best Sensitivity_Specificity threshold= 0.1 inc= 0.01"
## [1] "========================================"
## [1] "start= 0.09 end= 0.11 inc= 0.001"
## [1] "Thresh=0.09, Accuracy=93.5%, BaseAcc(Other)=93.8%, Sens=98.4%, Spec=93.2%, Sens^2+Spec^2=1.839"
## [1] "Thresh=0.091, Accuracy=93.5%, BaseAcc(Other)=93.8%, Sens=98.4%, Spec=93.2%, Sens^2+Spec^2=1.839
## [1] "Thresh=0.092, Accuracy=93.6%, BaseAcc(Other)=93.8%, Sens=98.4%, Spec=93.3%, Sens^2+Spec^2=1.839
## [1] "Thresh=0.093, Accuracy=93.6%, BaseAcc(Other)=93.8%, Sens=98.4%, Spec=93.3%, Sens^2+Spec^2=1.839
## [1] "Thresh=0.094, Accuracy=93.6%, BaseAcc(Other)=93.8%, Sens=98.3%, Spec=93.3%, Sens^2+Spec^2=1.839
## [1] "Thresh=0.095, Accuracy=93.7%, BaseAcc(Other)=93.8%, Sens=98.3%, Spec=93.4%, Sens^2+Spec^2=1.839
## [1] "Thresh=0.096, Accuracy=93.7%, BaseAcc(Other)=93.8%, Sens=98.3%, Spec=93.4%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.097, Accuracy=93.7%, BaseAcc(Other)=93.8%, Sens=98.3%, Spec=93.4%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.098, Accuracy=93.8%, BaseAcc(Other)=93.8%, Sens=98.2%, Spec=93.5%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.099, Accuracy=93.8%, BaseAcc(Other)=93.8%, Sens=98.2%, Spec=93.5%, Sens^2+Spec^2=1.841
## [1] "Thresh=0.1, Accuracy=93.8%, BaseAcc(Other)=93.8%, Sens=98.2%, Spec=93.5%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.101, Accuracy=93.9%, BaseAcc(Other)=93.8%, Sens=98.1%, Spec=93.6%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.102, Accuracy=93.9%, BaseAcc(Other)=93.8%, Sens=98.1%, Spec=93.6%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.103, Accuracy=93.9%, BaseAcc(Other)=93.8%, Sens=98.1%, Spec=93.6%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.104, Accuracy=93.9%, BaseAcc(Other)=93.8%, Sens=98.1%, Spec=93.7%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.105, Accuracy=94%, BaseAcc(Other)=93.8%, Sens=98%, Spec=93.7%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.106, Accuracy=94%, BaseAcc(Other)=93.8%, Sens=98%, Spec=93.7%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.107, Accuracy=94%, BaseAcc(Other)=93.8%, Sens=98%, Spec=93.7%, Sens^2+Spec^2=1.84"
## [1] "Thresh=0.108, Accuracy=94%, BaseAcc(Other)=93.8%, Sens=97.9%, Spec=93.8%, Sens^2+Spec^2=1.839"
## [1] "Thresh=0.109, Accuracy=94.1%, BaseAcc(Other)=93.8%, Sens=97.9%, Spec=93.8%, Sens^2+Spec^2=1.839
## [1] "========================================"
## [1] "Best Threshold=0.099"
## [1] "Best Sensitivity_Specificity=1.8410122788654"
```

```r
curThresh = as.numeric(result[bestThreshIndex])
Ponder_Ind_All_threshold = curThresh
```

The accuracy for the best threshold on the training set for Ponderosa Pine using all individuated data is shown below.

```r
result = calcLogisticModelAccuracy (forestTrain$PonderosaPine, Ponder_Ind_Train_predict,
                       curThresh, curThresh, 1, "Ponderosa", "Other", 3)
```

```
## [1] "Model Performance for threshold= 0.099"
## [1] "predicted performance="
##                        Predicted
## Actual                  FALSE=Predict:Other TRUE=Predict:Ponderosa
##   0=Actual:Other            357110 (TN)         24571 (FP)
##   1=Actual:Ponderosa        434 (FN)            24594 (TP)
## [1] "Sensitivity= 0.982659421447978 (True positive rate of Ponderosa = TP/(TP+FN) = 24594 /( 24594 +
## [1] "Specificity= 0.935624251665658 (True negative rate of Other = TN/(TN+FP) = 357110 /( 357110 + 24
## [1] "Sens^2+Spec^2=1.841"
## [1] "Baseline (Other) Accuracy=0.938462"
## [1] "Logistic Accuracy=0.938518"
```

The accuracy for the best threshold on the testing set for Ponderosa Pine using all individuated data is shown below.

```
result = calcLogisticModelAccuracy (forestTest$PonderosaPine, Ponder_Ind_Test_predict,
                        curThresh, curThresh, 1, "Ponderosa", "Other", 3,
                        saveFile=saveFileName, desc="Ponderosa All Individualized Vars",
                        AIC=Ponder_Ind_All_aic, AUC=Ponder_Ind_All_ROC_AUC)
```

```
## [1] "Model Performance for threshold= 0.099"
## [1] "predicted performance="
##                       Predicted
## Actual                 FALSE=Predict:Other TRUE=Predict:Ponderosa
##   0=Actual:Other           153003 (TN)          10574 (FP)
##   1=Actual:Ponderosa       202 (FN)             10524 (TP)
## [1] "Sensitivity= 0.981167257132202 (True positive rate of Ponderosa = TP/(TP+FN) = 10524 /( 10524 +
## [1] "Specificity= 0.935357660306767 (True negative rate of Other = TN/(TN+FP) = 153003 /( 153003 + 1
## [1] "Sens^2+Spec^2=1.837"
## [1] "Baseline (Other) Accuracy=0.938463"
## [1] "Logistic Accuracy=0.938176"
```

```
list[RC, Ponder_Ind_All_model_acc, Ponder_Ind_All_baseline_acc,
      TN, FN, FP, TP, Ponder_Ind_All_sens, Ponder_Ind_All_spec] <- result
  if (RC != "OK") {
    print(paste("Error - terminating:",RC))
    knitr:knit_exit()
  }
  Ponder_Ind_All_model_acc = as.integer(as.numeric(Ponder_Ind_All_model_acc)*1000)/10
  Ponder_Ind_All_baseline_acc = as.integer(as.numeric(Ponder_Ind_All_baseline_acc)*1000)/10
  Ponder_Ind_All_sens = as.integer(as.numeric(Ponder_Ind_All_sens)*1000)/10
  Ponder_Ind_All_spec = as.integer(as.numeric(Ponder_Ind_All_spec)*1000)/10
```

The Ponderosa Pine aggregated model accuracy on the test data is 77.15% compared to 77.12% for the individuated data model, essentially identical. Both are ~ 14% better than the baseline model.

Ponderosa Pine Logistic Regression - Significant Variables

Create Ponderosa Pine Logistic Model - Sig Vars

Now create the logistic model for the Aggregated Soil data using just the significant variables and compare to the previous models.

Ponderosa Pine Logistic Model using Significant Aggregated Data

Variables that have been removed are commented out in the code below.

```
  Ponder_Agg_LogMod =
    glm(PonderosaPine ~
          Elev +     # Elevation in meters of cell
          Aspect +   # Direction in degrees slope faces
          Slope +    # Slope / steepness of hill in degrees (0 to 90)
          H2OHD +    # Horizontal distance in meters to nearest water
          H2OVD +    # Vertical distance in meters to nearest water
          RoadHD +   # Horizontal distance in meters to nearest road
          FirePtHD + # Horizontal distance in meters to nearest fire point
          Shade9AM + Shade12PM + Shade3PM + # Amount of shade at 9am, 12pm and 3pm
```

```r
        # Wilderness areas:
          # RWwild + NEwild +
          CMwild
          # CPwild +
        # Aggregated Soil type:
          # 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
          ,
        data=forestTrain, family=binomial)

# save model for later use
Ponder_Agg_Sig_LogMod = Ponder_Agg_LogMod
save("Ponder_Agg_Sig_LogMod", file="Ponder_Agg_Sig_LogMod.Rdata")

Ponder_Agg_Sig_aic<-as.integer(Ponder_Agg_LogMod$aic)
Ponder_Agg_Sig_aic
```

```
## [1] 77296
```

Check the coefficients of the Ponderosa Pine model using significant aggregated data.

```r
summary(Ponder_Agg_LogMod)
```

```
##
## Call:
## glm(formula = PonderosaPine ~ Elev + Aspect + Slope + H2OHD +
##     H2OVD + RoadHD + FirePtHD + Shade9AM + Shade12PM + Shade3PM +
##     CMwild, family = binomial, data = forestTrain)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -3.05959  -0.12153  -0.04307  -0.01694   3.05375
##
## Coefficients:
##               Estimate Std. Error  z value Pr(>|z|)
## (Intercept)  2.055e+01  5.055e-01   40.650  < 2e-16 ***
## Elev        -1.048e-02  7.140e-05 -146.746  < 2e-16 ***
## Aspect       1.629e-03  1.222e-04   13.323  < 2e-16 ***
## Slope        1.834e-02  2.981e-03    6.152 7.63e-10 ***
## H2OHD        2.651e-03  7.413e-05   35.766  < 2e-16 ***
## H2OVD        3.821e-03  2.021e-04   18.906  < 2e-16 ***
## RoadHD      -9.102e-05  1.497e-05   -6.078 1.21e-09 ***
## FirePtHD    -3.461e-04  1.547e-05  -22.375  < 2e-16 ***
## Shade9AM    -2.069e-02  2.848e-03   -7.264 3.76e-13 ***
```

```
## Shade12PM    5.280e-02  2.344e-03   22.528  < 2e-16 ***
## Shade3PM    -2.962e-02  2.347e-03  -12.621  < 2e-16 ***
## CMwild       1.606e+00  2.569e-02   62.527  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 188044  on 406708  degrees of freedom
## Residual deviance:  77273  on 406697  degrees of freedom
## AIC: 77297
##
## Number of Fisher Scoring iterations: 8
```

The intercept looks much more reasonable. Some soil types that were significant previously are no longer significant.

Ponderosa Pine Logistic Model using Significant Individuated Data

Create a logistic model for the significant individuated variables.

Again, the non-significant variables have been commented out.

```r
Ponder_Ind_LogMod =
  glm(PonderosaPine ~
        Elev +     # Elevation in meters of cell
        Aspect +   # Direction in degrees slope faces
        Slope +    # Slope / steepness of hill in degrees (0 to 90)
        H2OHD +    # Horizontal distance in meters to nearest water
        H2OVD +    # Vertical distance in meters to nearest water
        RoadHD +   # Horizontal distance in meters to nearest road
        FirePtHD + # Horizontal distance in meters to nearest fire point
        Shade9AM + Shade12PM + Shade3PM + # Amount of shade at 9am, 12pm and 3pm
        # Wilderness areas:
          # RWwild + NEwild +
          CMwild +
          # CPwild +
        # Climate Zone:
        # ClimateName +
          # Montane_low + Montane +
          # SubAlpine + Alpine +
          # Dry + Non_Dry +
        # Geology Zone:
        # GeoName +
          # Alluvium + Glacial +
          # Sed_mix + Ign_Meta +
        # Soil Family:
          # Aquolis_cmplx +
          # Argiborolis_Pachic +
          # Borohemists_cmplx + Bross +
          # Bullwark + Bullwark_Cmplx + Catamount + Catamount_cmplx +
          Cathedral +
          # Como +
          # Cryaquepts_cmplx + Cryaquepts_Typic + Cryaquolls +
          # Cryaquolls_cmplx + Cryaquolls_Typic + Cryaquolls_Typic_cmplx +
```

```r
          # Cryoborolis_cmplx +
          # Cryorthents +
          # Cryorthents_cmplx + Cryumbrepts + Cryumbrepts_cmplx + Gateview +
          # Gothic + Granile +
          Haploborolis +
          # Legault +
          # Legault_cmplx +
          # Leighcan + Leighcan_cmplx + Leighcan_warm +
          # Moran +
          Ratake +
          # Ratake_cmplx + Rogert + Supervisor_Limber_cmplx +
          # Troutville + Unspecified + Vanet +
          Wetmore
        # Soil Rock composition:
          # Bouldery_ext +
          # Rock_Land +
          # Rock_Land_cmplx + Rock_Outcrop +
          # Rock_Outcrop_cmplx +
          # Rubbly + Stony + Stony_extreme + Stony_very + Till_Substratum
          ,
        data=forestTrain, family=binomial)

# save model for later use
Ponder_Ind_Sig_LogMod = Ponder_Ind_LogMod
save("Ponder_Ind_Sig_LogMod", file="Ponder_Ind_Sig_LogMod.Rdata")

Ponder_Ind_Sig_aic<-as.integer(Ponder_Ind_LogMod$aic)
Ponder_Ind_Sig_aic
```

```
## [1] 71711
```

```r
summary(Ponder_Ind_LogMod)
```

```
##
## Call:
## glm(formula = PonderosaPine ~ Elev + Aspect + Slope + H2OHD +
##     H2OVD + RoadHD + FirePtHD + Shade9AM + Shade12PM + Shade3PM +
##     CMwild + Cathedral + Haploborolis + Ratake + Wetmore, family = binomial,
##     data = forestTrain)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -3.13999  -0.10743  -0.03966  -0.01621   2.96120
##
## Coefficients:
##               Estimate Std. Error  z value Pr(>|z|)
## (Intercept)  2.335e+01  5.328e-01   43.821  < 2e-16 ***
## Elev        -1.010e-02  7.687e-05 -131.382  < 2e-16 ***
## Aspect       1.538e-03  1.246e-04   12.343  < 2e-16 ***
## Slope       -2.642e-03  3.242e-03   -0.815   0.4152
## H2OHD        3.058e-03  7.934e-05   38.542  < 2e-16 ***
## H2OVD        2.392e-03  2.157e-04   11.087  < 2e-16 ***
## RoadHD      -8.696e-05  1.601e-05   -5.432 5.57e-08 ***
## FirePtHD    -4.025e-04  1.697e-05  -23.714  < 2e-16 ***
```

```
## Shade9AM     -2.699e-02  2.977e-03   -9.065  < 2e-16 ***
## Shade12PM     4.257e-02  2.485e-03   17.133  < 2e-16 ***
## Shade3PM     -3.009e-02  2.459e-03  -12.236  < 2e-16 ***
## CMwild        1.474e+00  2.886e-02   51.058  < 2e-16 ***
## Cathedral    -5.267e-01  6.383e-02   -8.252  < 2e-16 ***
## Haploborolis -1.252e-01  4.882e-02   -2.564   0.0103 *
## Ratake        2.066e+00  3.509e-02   58.883  < 2e-16 ***
## Wetmore       1.386e+00  4.007e-02   34.594  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 188044  on 406708  degrees of freedom
## Residual deviance:  71679  on 406693  degrees of freedom
## AIC: 71711
##
## Number of Fisher Scoring iterations: 9
```

Again the intercept looks much better. Also a few variables have become non-significant.

Predict Ponderosa Pine Logistic Model Probabilities - Sig Vars

Ponderosa Pine Probabilities using Significant Aggregated Data

Predict the probability of Ponderosa Pine for aggregated Data - significant variables.

```r
# Predict Ponderosa Pine Agg Data - significant variables

  Ponder_Agg_Train_predict= predict(Ponder_Agg_LogMod, type="response")
  summary(Ponder_Agg_Train_predict)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000002 0.0002375 0.0014250 0.0615379 0.0145701 0.9998007
```

```r
  Ponder_Agg_Test_predict= predict(Ponder_Agg_LogMod, type="response",newdata=forestTest)
  summary(Ponder_Agg_Test_predict)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000003 0.0002345 0.0014081 0.0615143 0.0145231 0.9999065
```

Ponderosa Pine Probabilities using Significant Individuated Data

Predict the probability of Ponderosa Pine using significant Individuated Data.

```r
  Ponder_Ind_Train_predict= predict(Ponder_Ind_LogMod, type="response")
  summary(Ponder_Ind_Train_predict)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000006 0.0002114 0.0011884 0.0615379 0.0112423 0.9994816
```

```r
  Ponder_Ind_Test_predict= predict(Ponder_Ind_LogMod, type="response",newdata=forestTest)
  summary(Ponder_Ind_Test_predict)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000004 0.0002097 0.0011758 0.0616537 0.0112369 0.9997315
```

```r
print(paste("ROCR graph 2 completed at",curTime))
```

```
## [1] "ROCR graph 2 completed at 2018-08-12 21:08:50"
```

Ponderosa Pine Receiver Operating Characteristic (ROC) - Sig Vars

Look at the True Positive and False Positive rates based on threshold value.

```r
if (calcROC) {
  ROCpred_Ponder_Agg = prediction(Ponder_Agg_Train_predict, forestTrain$PonderosaPine)
  summary(ROCpred_Ponder_Agg)

  ROCperf_Ponder_Agg = performance(ROCpred_Ponder_Agg, "tpr", "fpr")
  summary(ROCperf_Ponder_Agg)

  Ponder_Agg_Sig_ROC_AUC = as.numeric(performance(ROCpred_Ponder_Agg, "auc")@y.values)
  Ponder_Agg_Sig_ROC_AUC=as.integer(as.numeric(Ponder_Agg_Sig_ROC_AUC)*1000)/10
  Ponder_Agg_Sig_ROC_AUC

  jpeg(filename="Fig-ROCR_perf_Ponder_Agg_Sig.jpg")
  plot(ROCperf_Ponder_Agg, colorize=TRUE, print.cutoffs.at=seq(0,1,0.1), text.adj=c(-0.2,1.7))
  dev.off()
} else {
  Ponder_Agg_Sig_ROC_AUC = 83.7
}
```

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

```r
if (calcROC) {
  curTime=Sys.time()
  print(paste("ROCR graph 2 started at",curTime))

  ROCpred_Ponder_Ind = prediction(Ponder_Ind_Train_predict, forestTrain$PonderosaPine)
  summary(ROCpred_Ponder_Ind)

  ROCperf_Ponder_Ind = performance(ROCpred_Ponder_Ind, "tpr", "fpr")
  summary(ROCperf_Ponder_Ind)

  Ponder_Ind_Sig_ROC_AUC = as.numeric(performance(ROCpred_Ponder_Ind, "auc")@y.values)
  Ponder_Ind_Sig_ROC_AUC=as.integer(as.numeric(Ponder_Ind_Sig_ROC_AUC)*1000)/10
  Ponder_Ind_Sig_ROC_AUC

  jpeg(filename="Fig-ROC_perf_Ponder_Ind_Sig.jpg")
  plot(ROCperf_Ponder_Ind, colorize=TRUE, print.cutoffs.at=seq(0,1,0.1), text.adj=c(-0.2,1.7))
  dev.off()
} else {
  Ponder_Ind_Sig_ROC_AUC = 83.8
}
```

```
## [1] "ROCR graph 2 started at 2018-08-12 21:15:20"
```

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

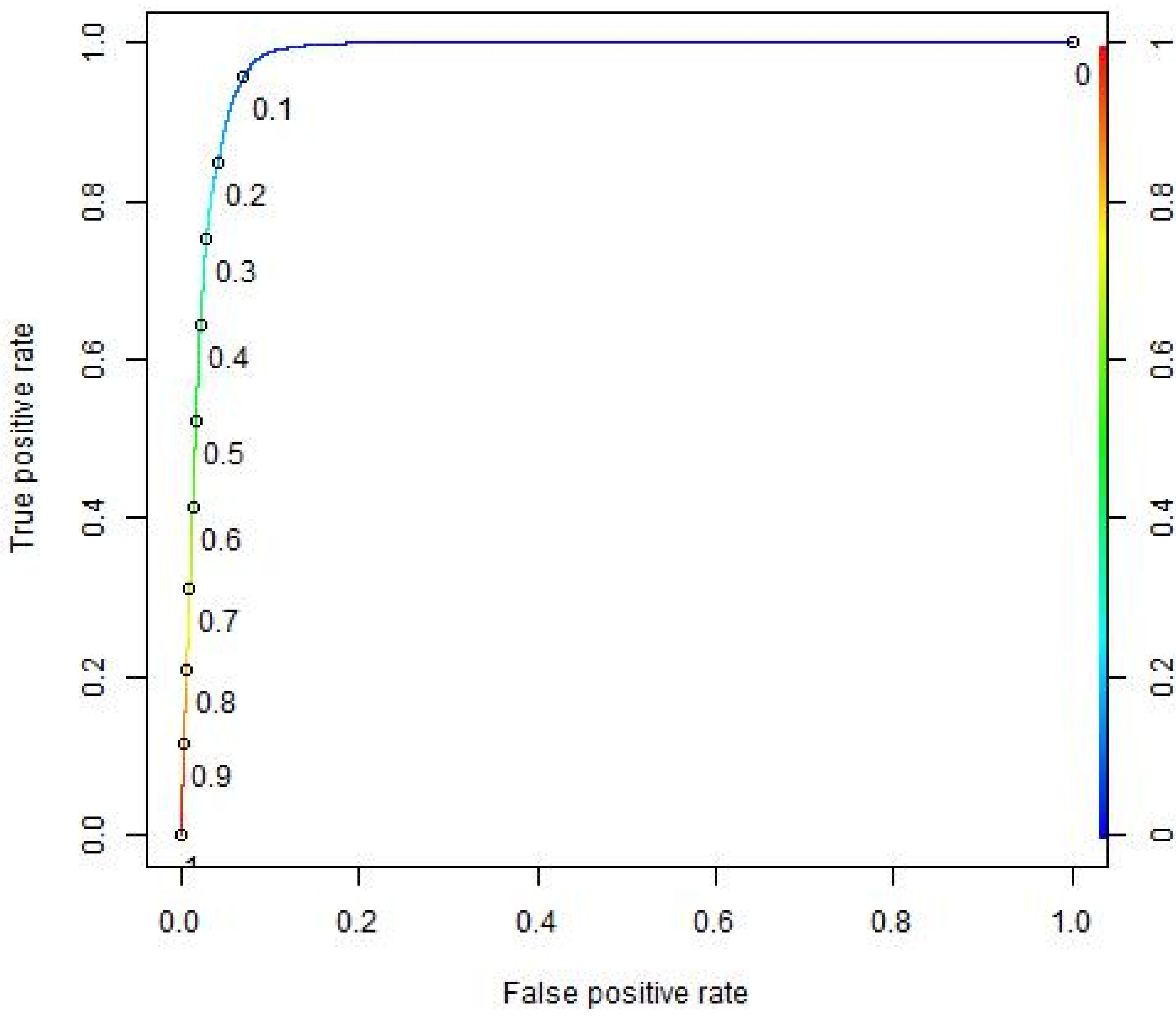

Figure 3: Ponderosa Pine ROC for Aggregated Significant Data

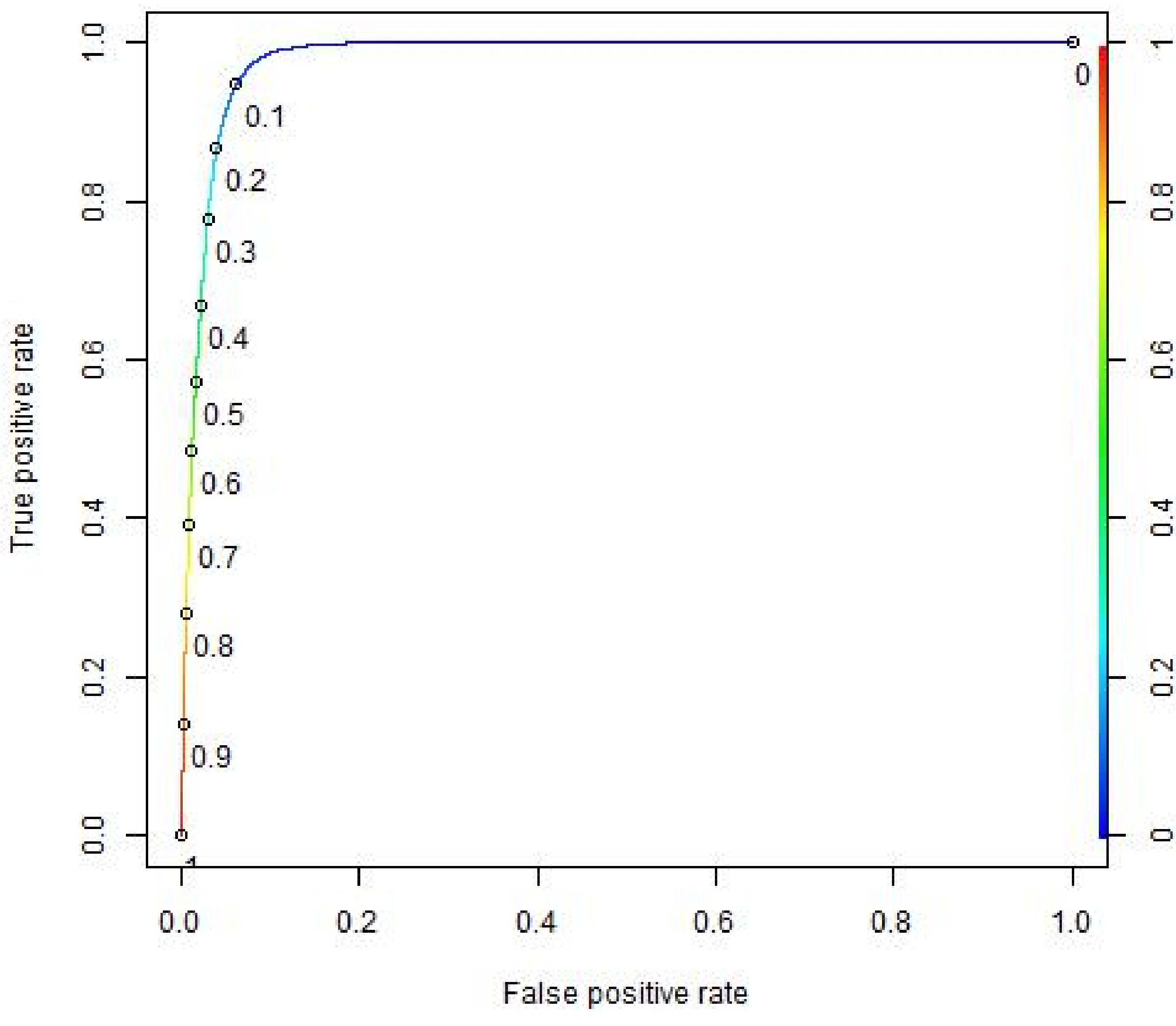

Figure 4: Ponderosa Pine ROC for Individuated Significant Data

The threshold graphs are essentially identical. This is making me think that there is not much difference between the two models. The AIC score for the Soil Type model is AIC: 351676 and for the individuated variables is: AIC: 351839. The Soil type model AIC score is 0.046% better than the individuated model.

Calculate Accuracy of Ponderosa Pine Logisitic Model - Sig Vars

Calculate Ponderosa Pine Aggregated Data Logisitic Model Accuracy - Significant Vars

Find best Ponderosa Pine threshold for Aggregated Data using significant variables.

```r
result = calcLogisticModelAccuracy (forestTrain$PonderosaPine, Ponder_Agg_Train_predict,
                        0.0, 1, 10, "Ponderosa", "Other", 1,1)
```

```
## [1] "Searching for threshold producing best Sensitivity_Specificity"
## [1] "start= 0 end= 1 inc= 0.1"
## [1] "Thresh=0, Accuracy=6.1%, BaseAcc(Other)=93.8%, Sens=100%, Spec=0%, Sens^2+Spec^2=-2"
## [1] "Thresh=0.1, Accuracy=93.1%, BaseAcc(Other)=93.8%, Sens=95.7%, Spec=92.9%, Sens^2+Spec^2=1.781"
## [1] "Thresh=0.2, Accuracy=95.2%, BaseAcc(Other)=93.8%, Sens=84.7%, Spec=95.9%, Sens^2+Spec^2=1.638"
## [1] "Thresh=0.3, Accuracy=95.8%, BaseAcc(Other)=93.8%, Sens=75.3%, Spec=97.2%, Sens^2+Spec^2=1.513"
## [1] "Thresh=0.4, Accuracy=95.8%, BaseAcc(Other)=93.8%, Sens=64.2%, Spec=97.8%, Sens^2+Spec^2=1.371"
## [1] "Thresh=0.5, Accuracy=95.5%, BaseAcc(Other)=93.8%, Sens=52.3%, Spec=98.3%, Sens^2+Spec^2=1.241"
## [1] "Thresh=0.6, Accuracy=95.2%, BaseAcc(Other)=93.8%, Sens=41.4%, Spec=98.7%, Sens^2+Spec^2=1.146"
## [1] "Thresh=0.7, Accuracy=94.8%, BaseAcc(Other)=93.8%, Sens=31.2%, Spec=99%, Sens^2+Spec^2=1.079"
## [1] "Thresh=0.8, Accuracy=94.5%, BaseAcc(Other)=93.8%, Sens=20.8%, Spec=99.4%, Sens^2+Spec^2=1.031"
## [1] "Thresh=0.9, Accuracy=94.2%, BaseAcc(Other)=93.8%, Sens=11.6%, Spec=99.7%, Sens^2+Spec^2=1.007"
## [1] "Thresh=1, Accuracy=93.8%, BaseAcc(Other)=93.8%, Sens=0%, Spec=100%, Sens^2+Spec^2=-2"
## [1] "Best Sensitivity_Specificity threshold= 0.1 inc= 0.1"
## [1] "========================================="
## [1] "start= 0 end= 0.2 inc= 0.01"
## [1] "Thresh=0, Accuracy=6.1%, BaseAcc(Other)=93.8%, Sens=100%, Spec=0%, Sens^2+Spec^2=-2"
## [1] "Thresh=0.01, Accuracy=77.9%, BaseAcc(Other)=93.8%, Sens=99.9%, Spec=76.5%, Sens^2+Spec^2=1.585"
## [1] "Thresh=0.02, Accuracy=83.4%, BaseAcc(Other)=93.8%, Sens=99.9%, Spec=82.3%, Sens^2+Spec^2=1.677"
## [1] "Thresh=0.03, Accuracy=86.2%, BaseAcc(Other)=93.8%, Sens=99.7%, Spec=85.3%, Sens^2+Spec^2=1.722"
## [1] "Thresh=0.04, Accuracy=88%, BaseAcc(Other)=93.8%, Sens=99.4%, Spec=87.3%, Sens^2+Spec^2=1.752"
## [1] "Thresh=0.05, Accuracy=89.4%, BaseAcc(Other)=93.8%, Sens=99.1%, Spec=88.8%, Sens^2+Spec^2=1.772"
## [1] "Thresh=0.06, Accuracy=90.5%, BaseAcc(Other)=93.8%, Sens=98.8%, Spec=90%, Sens^2+Spec^2=1.787"
## [1] "Thresh=0.07, Accuracy=91.4%, BaseAcc(Other)=93.8%, Sens=98.2%, Spec=90.9%, Sens^2+Spec^2=1.792"
## [1] "Thresh=0.08, Accuracy=92.1%, BaseAcc(Other)=93.8%, Sens=97.7%, Spec=91.7%, Sens^2+Spec^2=1.796"
## [1] "Thresh=0.09, Accuracy=92.6%, BaseAcc(Other)=93.8%, Sens=96.8%, Spec=92.3%, Sens^2+Spec^2=1.791"
## [1] "Thresh=0.1, Accuracy=93.1%, BaseAcc(Other)=93.8%, Sens=95.7%, Spec=92.9%, Sens^2+Spec^2=1.781"
## [1] "Thresh=0.11, Accuracy=93.5%, BaseAcc(Other)=93.8%, Sens=94.6%, Spec=93.4%, Sens^2+Spec^2=1.77"
## [1] "Thresh=0.12, Accuracy=93.9%, BaseAcc(Other)=93.8%, Sens=93.6%, Spec=93.9%, Sens^2+Spec^2=1.758"
## [1] "Thresh=0.13, Accuracy=94.1%, BaseAcc(Other)=93.8%, Sens=92.5%, Spec=94.2%, Sens^2+Spec^2=1.745"
## [1] "Thresh=0.14, Accuracy=94.4%, BaseAcc(Other)=93.8%, Sens=91.4%, Spec=94.6%, Sens^2+Spec^2=1.731"
## [1] "Thresh=0.15, Accuracy=94.5%, BaseAcc(Other)=93.8%, Sens=90.2%, Spec=94.8%, Sens^2+Spec^2=1.715"
## [1] "Thresh=0.16, Accuracy=94.7%, BaseAcc(Other)=93.8%, Sens=89.1%, Spec=95.1%, Sens^2+Spec^2=1.699"
## [1] "Thresh=0.17, Accuracy=94.9%, BaseAcc(Other)=93.8%, Sens=88%, Spec=95.3%, Sens^2+Spec^2=1.684"
## [1] "Thresh=0.18, Accuracy=95%, BaseAcc(Other)=93.8%, Sens=86.8%, Spec=95.5%, Sens^2+Spec^2=1.667"
## [1] "Thresh=0.19, Accuracy=95.1%, BaseAcc(Other)=93.8%, Sens=85.7%, Spec=95.7%, Sens^2+Spec^2=1.651"
## [1] "Best Sensitivity_Specificity threshold= 0.08 inc= 0.01"
## [1] "========================================="
## [1] "start= 0.07 end= 0.09 inc= 0.001"
## [1] "Thresh=0.07, Accuracy=91.4%, BaseAcc(Other)=93.8%, Sens=98.2%, Spec=90.9%, Sens^2+Spec^2=1.792"
## [1] "Thresh=0.071, Accuracy=91.4%, BaseAcc(Other)=93.8%, Sens=98.2%, Spec=91%, Sens^2+Spec^2=1.793"
## [1] "Thresh=0.072, Accuracy=91.5%, BaseAcc(Other)=93.8%, Sens=98.1%, Spec=91.1%, Sens^2+Spec^2=1.793
```

```
## [1] "Thresh=0.073, Accuracy=91.6%, BaseAcc(Other)=93.8%, Sens=98%, Spec=91.2%, Sens^2+Spec^2=1.794"
## [1] "Thresh=0.074, Accuracy=91.7%, BaseAcc(Other)=93.8%, Sens=98%, Spec=91.2%, Sens^2+Spec^2=1.794"
## [1] "Thresh=0.075, Accuracy=91.7%, BaseAcc(Other)=93.8%, Sens=98%, Spec=91.3%, Sens^2+Spec^2=1.795"
## [1] "Thresh=0.076, Accuracy=91.8%, BaseAcc(Other)=93.8%, Sens=97.9%, Spec=91.4%, Sens^2+Spec^2=1.795
## [1] "Thresh=0.077, Accuracy=91.9%, BaseAcc(Other)=93.8%, Sens=97.8%, Spec=91.5%, Sens^2+Spec^2=1.795
## [1] "Thresh=0.078, Accuracy=91.9%, BaseAcc(Other)=93.8%, Sens=97.8%, Spec=91.5%, Sens^2+Spec^2=1.795
## [1] "Thresh=0.079, Accuracy=92%, BaseAcc(Other)=93.8%, Sens=97.7%, Spec=91.6%, Sens^2+Spec^2=1.796"
## [1] "Thresh=0.08, Accuracy=92.1%, BaseAcc(Other)=93.8%, Sens=97.7%, Spec=91.7%, Sens^2+Spec^2=1.796"
## [1] "Thresh=0.081, Accuracy=92.1%, BaseAcc(Other)=93.8%, Sens=97.6%, Spec=91.8%, Sens^2+Spec^2=1.796
## [1] "Thresh=0.082, Accuracy=92.2%, BaseAcc(Other)=93.8%, Sens=97.5%, Spec=91.8%, Sens^2+Spec^2=1.796
## [1] "Thresh=0.083, Accuracy=92.2%, BaseAcc(Other)=93.8%, Sens=97.4%, Spec=91.9%, Sens^2+Spec^2=1.795
## [1] "Thresh=0.084, Accuracy=92.3%, BaseAcc(Other)=93.8%, Sens=97.4%, Spec=92%, Sens^2+Spec^2=1.795"
## [1] "Thresh=0.085, Accuracy=92.4%, BaseAcc(Other)=93.8%, Sens=97.3%, Spec=92%, Sens^2+Spec^2=1.795"
## [1] "Thresh=0.086, Accuracy=92.4%, BaseAcc(Other)=93.8%, Sens=97.2%, Spec=92.1%, Sens^2+Spec^2=1.794
## [1] "Thresh=0.087, Accuracy=92.5%, BaseAcc(Other)=93.8%, Sens=97.1%, Spec=92.1%, Sens^2+Spec^2=1.793
## [1] "Thresh=0.088, Accuracy=92.5%, BaseAcc(Other)=93.8%, Sens=97%, Spec=92.2%, Sens^2+Spec^2=1.792"
## [1] "Thresh=0.089, Accuracy=92.6%, BaseAcc(Other)=93.8%, Sens=96.9%, Spec=92.3%, Sens^2+Spec^2=1.792
## [1] "========================================"
## [1] "Best Threshold=0.082"
## [1] "Best Sensitivity_Specificity=1.79631206204131"
```

```
curThresh = as.numeric(result[bestThreshIndex])
Ponder_Agg_Sig_threshold = curThresh
```

The accuracy for the best threshold on the training set for Ponderosa Pine using significant aggregated data is shown below.

```
result = calcLogisticModelAccuracy (forestTrain$PonderosaPine, Ponder_Agg_Train_predict,
                       curThresh, curThresh, 1, "Ponderosa", "Other", 3)
```

```
## [1] "Model Performance for threshold= 0.082"
## [1] "predicted performance="
##                       Predicted
## Actual                 FALSE=Predict:Other TRUE=Predict:Ponderosa
##   0=Actual:Other          350696 (TN)         30985 (FP)
##   1=Actual:Ponderosa      607 (FN)            24421 (TP)
## [1] "Sensitivity= 0.975747163177242 (True positive rate of Ponderosa = TP/(TP+FN) = 24421 /( 24421 +
## [1] "Specificity= 0.918819642581108 (True negative rate of Other = TN/(TN+FP) = 350696 /( 350696 + 3
## [1] "Sens^2+Spec^2=1.796"
## [1] "Baseline (Other) Accuracy=0.938462"
## [1] "Logistic Accuracy=0.922322"
```

The accuracy for the best threshold on the testing set for Ponderosa Pine using significant aggregated data is shown below.

```
result = calcLogisticModelAccuracy (forestTest$PonderosaPine, Ponder_Agg_Test_predict,
                       curThresh, curThresh, 1, "Ponderosa", "Other", 3,
                       saveFile=saveFileName, desc="Ponderosa Sig Aggregate Vars",
                       AIC=Ponder_Agg_Sig_aic, AUC=Ponder_Agg_Sig_ROC_AUC)
```

```
## [1] "Model Performance for threshold= 0.082"
## [1] "predicted performance="
##                       Predicted
## Actual                 FALSE=Predict:Other TRUE=Predict:Ponderosa
##   0=Actual:Other          150287 (TN)         13290 (FP)
##   1=Actual:Ponderosa      243 (FN)            10483 (TP)
## [1] "Sensitivity= 0.977344769718441 (True positive rate of Ponderosa = TP/(TP+FN) = 10483 /( 10483 +
```

```
## [1] "Specificity= 0.918753859038862 (True negative rate of Other = TN/(TN+FP) = 150287 /( 150287 + 13
## [1] "Sens^2+Spec^2=1.799"
## [1] "Baseline (Other) Accuracy=0.938463"
## [1] "Logistic Accuracy=0.922359"
```

```
list[RC, Ponder_Agg_Sig_model_acc, Ponder_Agg_Sig_baseline_acc,
      TN, FN, FP, TP, Ponder_Agg_Sig_sens, Ponder_Agg_Sig_spec] <- result
  if (RC != "OK") {
    print(paste("Error - terminating:",RC))
    knitr:knit_exit()
  }
  Ponder_Agg_Sig_model_acc = as.integer(as.numeric(Ponder_Agg_Sig_model_acc)*1000)/10
  Ponder_Agg_Sig_baseline_acc = as.integer(as.numeric(Ponder_Agg_Sig_baseline_acc)*1000)/10
  Ponder_Agg_Sig_sens = as.integer(as.numeric(Ponder_Agg_Sig_sens)*1000)/10
  Ponder_Agg_Sig_spec = as.integer(as.numeric(Ponder_Agg_Sig_spec)*1000)/10
```

Calculate Ponderosa Pine Individuated Data Logisitic Model Accuracy - Significant Vars

Find best Ponderosa Pine threshold for Inividuated Data using significant variables.

```
result = calcLogisticModelAccuracy (forestTrain$PonderosaPine, Ponder_Ind_Train_predict,
                        0.0, 1, 10, "Ponderosa", "Other", 1,1)
```

```
## [1] "Searching for threshold producing best Sensitivity_Specificity"
## [1] "start= 0 end= 1 inc= 0.1"
## [1] "Thresh=0, Accuracy=6.1%, BaseAcc(Other)=93.8%, Sens=100%, Spec=0%, Sens^2+Spec^2=-2"
## [1] "Thresh=0.1, Accuracy=93.9%, BaseAcc(Other)=93.8%, Sens=94.7%, Spec=93.8%, Sens^2+Spec^2=1.778"
## [1] "Thresh=0.2, Accuracy=95.4%, BaseAcc(Other)=93.8%, Sens=86.7%, Spec=96%, Sens^2+Spec^2=1.674"
## [1] "Thresh=0.3, Accuracy=95.8%, BaseAcc(Other)=93.8%, Sens=77.7%, Spec=97%, Sens^2+Spec^2=1.545"
## [1] "Thresh=0.4, Accuracy=95.9%, BaseAcc(Other)=93.8%, Sens=66.8%, Spec=97.8%, Sens^2+Spec^2=1.403"
## [1] "Thresh=0.5, Accuracy=95.8%, BaseAcc(Other)=93.8%, Sens=57.2%, Spec=98.3%, Sens^2+Spec^2=1.295"
## [1] "Thresh=0.6, Accuracy=95.7%, BaseAcc(Other)=93.8%, Sens=48.4%, Spec=98.8%, Sens^2+Spec^2=1.211"
## [1] "Thresh=0.7, Accuracy=95.4%, BaseAcc(Other)=93.8%, Sens=39.3%, Spec=99.1%, Sens^2+Spec^2=1.137"
## [1] "Thresh=0.8, Accuracy=95%, BaseAcc(Other)=93.8%, Sens=28.1%, Spec=99.4%, Sens^2+Spec^2=1.068"
## [1] "Thresh=0.9, Accuracy=94.5%, BaseAcc(Other)=93.8%, Sens=14%, Spec=99.7%, Sens^2+Spec^2=1.015"
## [1] "Thresh=1, Accuracy=93.8%, BaseAcc(Other)=93.8%, Sens=0%, Spec=100%, Sens^2+Spec^2=-2"
## [1] "Best Sensitivity_Specificity threshold= 0.1 inc= 0.1"
## [1] "========================================"
## [1] "start= 0 end= 0.2 inc= 0.01"
## [1] "Thresh=0, Accuracy=6.1%, BaseAcc(Other)=93.8%, Sens=100%, Spec=0%, Sens^2+Spec^2=-2"
## [1] "Thresh=0.01, Accuracy=80.2%, BaseAcc(Other)=93.8%, Sens=100%, Spec=78.9%, Sens^2+Spec^2=-2"
## [1] "Thresh=0.02, Accuracy=85.3%, BaseAcc(Other)=93.8%, Sens=99.8%, Spec=84.3%, Sens^2+Spec^2=1.708"
## [1] "Thresh=0.03, Accuracy=88%, BaseAcc(Other)=93.8%, Sens=99.4%, Spec=87.2%, Sens^2+Spec^2=1.75"
## [1] "Thresh=0.04, Accuracy=89.6%, BaseAcc(Other)=93.8%, Sens=99%, Spec=89%, Sens^2+Spec^2=1.774"
## [1] "Thresh=0.05, Accuracy=90.9%, BaseAcc(Other)=93.8%, Sens=98.5%, Spec=90.4%, Sens^2+Spec^2=1.79"
## [1] "Thresh=0.06, Accuracy=91.9%, BaseAcc(Other)=93.8%, Sens=97.7%, Spec=91.5%, Sens^2+Spec^2=1.794"
## [1] "Thresh=0.07, Accuracy=92.6%, BaseAcc(Other)=93.8%, Sens=96.9%, Spec=92.3%, Sens^2+Spec^2=1.793"
## [1] "Thresh=0.08, Accuracy=93.1%, BaseAcc(Other)=93.8%, Sens=96.2%, Spec=92.9%, Sens^2+Spec^2=1.789"
## [1] "Thresh=0.09, Accuracy=93.5%, BaseAcc(Other)=93.8%, Sens=95.5%, Spec=93.4%, Sens^2+Spec^2=1.785"
## [1] "Thresh=0.1, Accuracy=93.9%, BaseAcc(Other)=93.8%, Sens=94.7%, Spec=93.8%, Sens^2+Spec^2=1.778"
## [1] "Thresh=0.11, Accuracy=94.2%, BaseAcc(Other)=93.8%, Sens=93.8%, Spec=94.2%, Sens^2+Spec^2=1.768"
## [1] "Thresh=0.12, Accuracy=94.4%, BaseAcc(Other)=93.8%, Sens=92.9%, Spec=94.5%, Sens^2+Spec^2=1.757"
## [1] "Thresh=0.13, Accuracy=94.6%, BaseAcc(Other)=93.8%, Sens=92%, Spec=94.8%, Sens^2+Spec^2=1.745"
## [1] "Thresh=0.14, Accuracy=94.8%, BaseAcc(Other)=93.8%, Sens=91.1%, Spec=95%, Sens^2+Spec^2=1.734"
## [1] "Thresh=0.15, Accuracy=94.9%, BaseAcc(Other)=93.8%, Sens=90.4%, Spec=95.2%, Sens^2+Spec^2=1.725"
```

```
## [1] "Thresh=0.16, Accuracy=95%, BaseAcc(Other)=93.8%, Sens=89.7%, Spec=95.4%, Sens^2+Spec^2=1.715"
## [1] "Thresh=0.17, Accuracy=95.1%, BaseAcc(Other)=93.8%, Sens=88.9%, Spec=95.5%, Sens^2+Spec^2=1.704"
## [1] "Thresh=0.18, Accuracy=95.2%, BaseAcc(Other)=93.8%, Sens=88.1%, Spec=95.7%, Sens^2+Spec^2=1.693"
## [1] "Thresh=0.19, Accuracy=95.3%, BaseAcc(Other)=93.8%, Sens=87.4%, Spec=95.8%, Sens^2+Spec^2=1.684"
## [1] "Best Sensitivity_Specificity threshold= 0.06 inc= 0.01"
## [1] "========================================"
## [1] "start= 0.05 end= 0.07 inc= 0.001"
## [1] "Thresh=0.05, Accuracy=90.9%, BaseAcc(Other)=93.8%, Sens=98.5%, Spec=90.4%, Sens^2+Spec^2=1.79"
## [1] "Thresh=0.051, Accuracy=91.1%, BaseAcc(Other)=93.8%, Sens=98.4%, Spec=90.6%, Sens^2+Spec^2=1.79"
## [1] "Thresh=0.052, Accuracy=91.2%, BaseAcc(Other)=93.8%, Sens=98.3%, Spec=90.7%, Sens^2+Spec^2=1.791
## [1] "Thresh=0.053, Accuracy=91.3%, BaseAcc(Other)=93.8%, Sens=98.3%, Spec=90.8%, Sens^2+Spec^2=1.791
## [1] "Thresh=0.054, Accuracy=91.4%, BaseAcc(Other)=93.8%, Sens=98.2%, Spec=90.9%, Sens^2+Spec^2=1.792
## [1] "Thresh=0.055, Accuracy=91.5%, BaseAcc(Other)=93.8%, Sens=98.1%, Spec=91%, Sens^2+Spec^2=1.793"
## [1] "Thresh=0.056, Accuracy=91.6%, BaseAcc(Other)=93.8%, Sens=98%, Spec=91.1%, Sens^2+Spec^2=1.792"
## [1] "Thresh=0.057, Accuracy=91.7%, BaseAcc(Other)=93.8%, Sens=97.9%, Spec=91.2%, Sens^2+Spec^2=1.793
## [1] "Thresh=0.058, Accuracy=91.7%, BaseAcc(Other)=93.8%, Sens=97.9%, Spec=91.3%, Sens^2+Spec^2=1.794
## [1] "Thresh=0.059, Accuracy=91.8%, BaseAcc(Other)=93.8%, Sens=97.8%, Spec=91.4%, Sens^2+Spec^2=1.794
## [1] "Thresh=0.06, Accuracy=91.9%, BaseAcc(Other)=93.8%, Sens=97.7%, Spec=91.5%, Sens^2+Spec^2=1.794"
## [1] "Thresh=0.061, Accuracy=92%, BaseAcc(Other)=93.8%, Sens=97.6%, Spec=91.6%, Sens^2+Spec^2=1.793"
## [1] "Thresh=0.062, Accuracy=92.1%, BaseAcc(Other)=93.8%, Sens=97.5%, Spec=91.7%, Sens^2+Spec^2=1.793
## [1] "Thresh=0.063, Accuracy=92.1%, BaseAcc(Other)=93.8%, Sens=97.5%, Spec=91.8%, Sens^2+Spec^2=1.793
## [1] "Thresh=0.064, Accuracy=92.2%, BaseAcc(Other)=93.8%, Sens=97.4%, Spec=91.9%, Sens^2+Spec^2=1.794
## [1] "Thresh=0.065, Accuracy=92.3%, BaseAcc(Other)=93.8%, Sens=97.3%, Spec=91.9%, Sens^2+Spec^2=1.794
## [1] "Thresh=0.066, Accuracy=92.3%, BaseAcc(Other)=93.8%, Sens=97.3%, Spec=92%, Sens^2+Spec^2=1.794"
## [1] "Thresh=0.067, Accuracy=92.4%, BaseAcc(Other)=93.8%, Sens=97.2%, Spec=92.1%, Sens^2+Spec^2=1.794
## [1] "Thresh=0.068, Accuracy=92.5%, BaseAcc(Other)=93.8%, Sens=97.1%, Spec=92.2%, Sens^2+Spec^2=1.794
## [1] "Thresh=0.069, Accuracy=92.5%, BaseAcc(Other)=93.8%, Sens=97%, Spec=92.2%, Sens^2+Spec^2=1.794"
## [1] "========================================"
## [1] "Best Threshold=0.068"
## [1] "Best Sensitivity_Specificity=1.79496738350594"
```

```
curThresh = as.numeric(result[bestThreshIndex])
Ponder_Ind_Sig_threshold = curThresh
```

The accuracy for the best threshold on the training set for Ponderosa Pine using significant individuated data is shown below.

```
result = calcLogisticModelAccuracy (forestTrain$PonderosaPine, Ponder_Ind_Train_predict,
                        curThresh, curThresh, 1, "Ponderosa", "Other", 3)
```

```
## [1] "Model Performance for threshold= 0.068"
## [1] "predicted performance="
##                      Predicted
## Actual                FALSE=Predict:Other TRUE=Predict:Ponderosa
##   0=Actual:Other         351950 (TN)         29731 (FP)
##   1=Actual:Ponderosa     702 (FN)            24326 (TP)
## [1] "Sensitivity= 0.971951414415854 (True positive rate of Ponderosa = TP/(TP+FN) = 24326 /( 24326 +
## [1] "Specificity= 0.922105108716441 (True negative rate of Other = TN/(TN+FP) = 351950 /( 351950 + 29
## [1] "Sens^2+Spec^2=1.794"
## [1] "Baseline (Other) Accuracy=0.938462"
## [1] "Logistic Accuracy=0.925172"
```

The accuracy for the best threshold on the testing set for Ponderosa Pine using significant individuated data is shown below.

```
result = calcLogisticModelAccuracy (forestTest$PonderosaPine, Ponder_Ind_Test_predict,
                        curThresh, curThresh, 1, "Ponderosa", "Other", 3,
                        saveFile=saveFileName, desc="Ponderosa Sig Individualized Vars",
                        AIC=Ponder_Ind_Sig_aic, AUC=Ponder_Ind_Sig_ROC_AUC)
```

```
## [1] "Model Performance for threshold= 0.068"
## [1] "predicted performance="
##                      Predicted
## Actual                FALSE=Predict:Other TRUE=Predict:Ponderosa
##   0=Actual:Other          150766 (TN)         12811 (FP)
##   1=Actual:Ponderosa      299 (FN)            10427 (TP)
## [1] "Sensitivity= 0.972123811299646 (True positive rate of Ponderosa = TP/(TP+FN) = 10427 /( 10427 +
## [1] "Specificity= 0.921682143577642 (True negative rate of Other = TN/(TN+FP) = 150766 /( 150766 + 1
## [1] "Sens^2+Spec^2=1.794"
## [1] "Baseline (Other) Accuracy=0.938463"
## [1] "Logistic Accuracy=0.924786"
```

```
list[RC, Ponder_Ind_Sig_model_acc, Ponder_Ind_Sig_baseline_acc,
      TN, FN, FP, TP, Ponder_Ind_Sig_sens, Ponder_Ind_Sig_spec] <- result
  if (RC != "OK") {
    print(paste("Error - terminating:",RC))
    knitr:knit_exit()
  }
  Ponder_Ind_Sig_model_acc = as.integer(as.numeric(Ponder_Ind_Sig_model_acc)*1000)/10
  Ponder_Ind_Sig_baseline_acc = as.integer(as.numeric(Ponder_Ind_Sig_baseline_acc)*1000)/10
  Ponder_Ind_Sig_sens = as.integer(as.numeric(Ponder_Ind_Sig_sens)*1000)/10
  Ponder_Ind_Sig_spec = as.integer(as.numeric(Ponder_Ind_Sig_spec)*1000)/10

############# End End End End End End End End End End End End #################
```

The accuracy of the models is shown below:

Logistic Model	Accuracy	Sens	Spec	AIC	AUC	Threshhold
Ponderosa Pine Aggregate All Vars	93.8%	98.1%	93.5%	61850	98.3%	0.099
Ponderosa Pine Individual All Vars	93.8%	98.1%	93.5%	61856	98.3%	0.099
Ponderosa Pine Aggregate Sig Vars	92.2%	97.7%	91.8%	77296	97.8%	0.082
Ponderosa Pine Individual Sig Vars	92.4%	97.2%	92.1%	71711	98%	0.068

There is a slight degradation in the accuracy with insignificant variables eliminated, but not by much.

Conclusion

It is beginning to look like there is no advantage to dis-aggregating the Soil Type variables into their component parts. I was hoping there would be some improvement by allowing the individual variables to be “more finely” tuned. There is probably a mathematical explanation that proves there is no advantage of breaking out aggregated variables. I have to think about that more.

The logistic regression results for Spruce and Fir are 7% better than the original paper this project was modeled after. These tests need to be done for the remaining 6 forest cover types to see how regression does overall.

```r
curTime=Sys.time()
print(paste("Forest Cover Logistic script ended at",curTime))
```

```
## [1] "Forest Cover Logistic script ended at 2018-08-12 21:21:04"
```