edx Capstone - Forest Cover Type

Saigopal Sathyamurthy

20/7/2021

1. Introduction

This report is prepared for the edx Data Science: Capstone project - the final course in HarvardX's Data Science Professional Certificate series. Forest Cover Type Data Set available in the UCI Machine Learning Repository was chosen for this assignment (**https://archive.ics.uci.edu/ml/datasets/covertype**).

There are many definitions for forest. The most widely used is the one formulated by the Food and Agriculture Organization. Briefly, forest is a land with a ***tree canopy cover*** of more than 10 percent and area of more than 0.5 hectares [1]. Forest cover type information are useful for natural resource managers and policy makers. It helps in reforestation and conservation activities.

The unit of observation in this study was a 30 x 30 m (digital spatial data) raster cells. These were obtained from the US Geological Survey (USGS). Several independent variables were derived from this using Geographic Information System (GIS) based surface analysis and hillshading procedures. The dependent variable (Cover Type) was taken from the US Forest Service (USFS) inventory information which used aerial photography. Study area included four wilderness areas located in the Roosevelt National Forest of northern Colorado. Detailed description of the methods used for deriving the independent variables from raster data can be found in the paper published by the dataset donor [2].

There are 581012 rows/observations and 55 columns/variables in the dataset. The first five rows and columns are printed below. There are no headers in the data downloaded. There are no missing values.

```
dim(dat)
```

```
[1] 581012     55
```

```
dat[c(1:5), c(1:5)]
```

```
     V1  V2 V3  V4  V5
1: 2596  51  3 258   0
2: 2590  56  2 212  -6
3: 2804 139  9 268  65
4: 2785 155 18 242 118
5: 2595  45  2 153  -1
```

```
any(is.na(dat))
```

```
[1] FALSE
```

Key step performed in this analysis include

- Wrangling data for headers and description of variables
- Creating new variables and modifying existing variables
- Splitting of the dataset, Exploratory Data Analysis (EDA) and selection of variable to be included in prediction models
- Comparison of the performance metric of three machine learning techniques with those reported by the dataset donor (linear discriminant analysis and artificial neural network)

2. Methods and Analysis

This section details the variables in the dataset, creating new variable/modifying existing variable, splitting of data, exploratory data analysis and selection of variables to be included in the models.

2.1 Variables descritpion and wrangling for headers

Headers are not included in the data. Information about the variables were provided in an info document available in the UCI repository. The names, data type and measurement unit of the variables are printed below. The first 10 variables are named as printed. For this, the **str_split** function was used with white space as pattern and row 1 through 10 of the first column was indexed (See R script for more details).

```
 [1] "Elevation                               quantitative    meters               ..."
 [2] "Aspect                                  quantitative    azimuth              ..."
 [3] "Slope                                   quantitative    degrees              ..."
 [4] "Horizontal_Distance_To_Hydrology        quantitative    meters               ..."
 [5] "Vertical_Distance_To_Hydrology          quantitative    meters               ..."
 [6] "Horizontal_Distance_To_Roadways         quantitative    meters               ..."
 [7] "Hillshade_9am                           quantitative    0 to 255 index       ..."
 [8] "Hillshade_Noon                          quantitative    0 to 255 index       ..."
 [9] "Hillshade_3pm                           quantitative    0 to 255 index       ..."
[10] "Horizontal_Distance_To_Fire_Points      quantitative    meters               ..."
[11] "Wilderness_Area (4 binary columns)      qualitative     0 (absence) or 1 (pre..."
[12] "Soil_Type (40 binary columns)           qualitative     0 (absence) or 1 (pre..."
[13] "Cover_Type (7 types)                    integer         1 to 7               ..."
```

The four wilderness area in this study were Rawah, Neota, Comanche Peak and Cache la Poudre respectively. These variables are mutually exclusive (one observation can not be present in more than one area) . The column names assigned to these variables are printed below.

```
[1] "rawah"           "neota"           "comanche_peak"   "cache_la_poudre"
```

Study code, Ecological Land Unit (ELU) code and description of a few of the forty soil types recorded in this study are tabulated below.

Table 1: Soil Types Codes & Description

	Study_code	ELU_code	Description
1	1	2702	Cathedral family - Rock outcrop complex, extremely stony.
2	2	2703	Vanet - Ratake families complex, very stony.
7	7	3501	Gothic family.
8	8	3502	Supervisor - Limber families complex.

These forty binary variables are mutually exclusive and are named by attaching the ELU code as suffix to the term 'soil_type_'. A few examples below.

```
[1] "soil_type_2702" "soil_type_2703" "soil_type_2704" "soil_type_2705"
[5] "soil_type_2706" "soil_type_2717" "soil_type_3501" "soil_type_3502"
```

The first digit of the ELU code represent the climatic zone and the second digit represent the geologic zone (Table 2). The last two digit were unique to mapping units and have no significance on climatic and geologic zones/class. These soil types can therefore be represented by fewer variables. For example soil types 2702 through 2717 can be clubbed to a general 'soil types 27' (Section 2.2.3).

Table 2: ELU Soil Classifciation

Climatic Zones	Geologic Zones
1. lower montane dry	1. alluvium
2. lower montane	2. glacial
3. montane dry	3. shale
4. montane	4. sandstone
5. montane dry and montane	5. mixed sedimentary
6. montane and subalpine	6. unspecified in the USFS ELU Survey
7. subalpine	7. igneous and metamorphic
8. alpine	8. volcanic

2.2 Creating new variables/Modifying existing variables

Two variables in this dataset (Aspect & Slope) were measured in degree's. New variables are created to capture the information provided by these variables, as these kinds of measurements are the domain of circular statistics. Circular statistics deals with direction, angles and axes (**https://en.wikipedia.org/wiki/Directional_statistics**). These are explained below.

2.2.1 Slope

Slope refers to the angle, or grade, of an incline. It is typically expressed as percent (gradient) (**https://www.nwcg.gov/course/ffm/vert-horiz-and-slope/45-slope**). In this study, slope was provided in degree's. This was converted into percent slope.

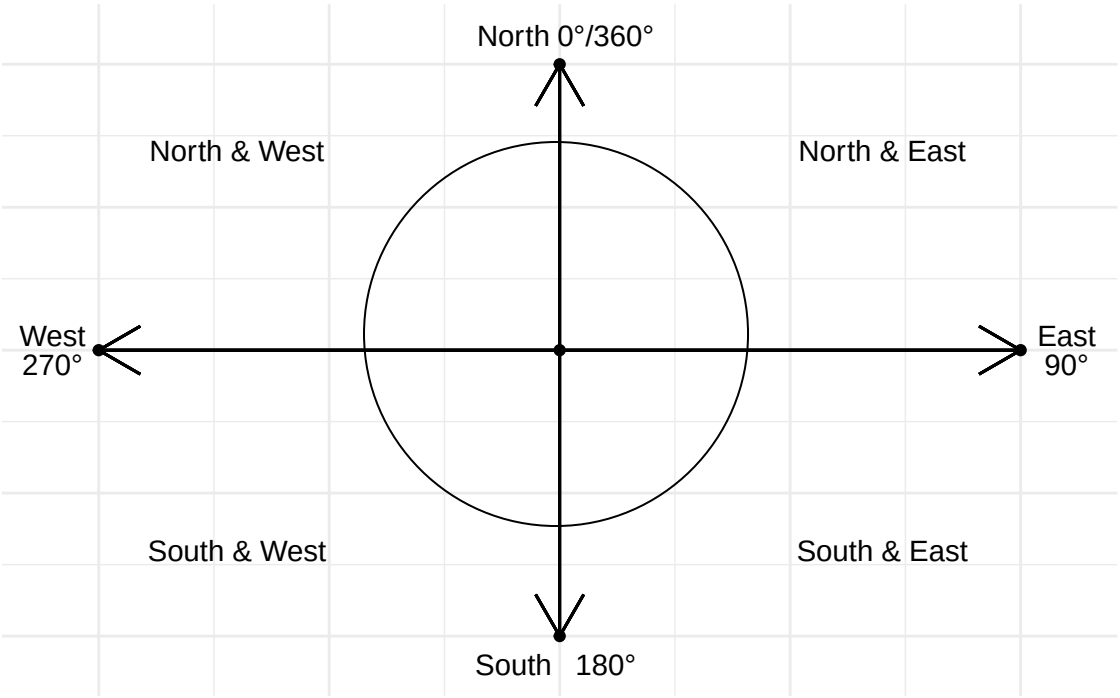
Percent slope is the tangent value of the angle in degree multiplied by 100. As the *tan* function in R takes input in radian, the degree was converted to radian before computing the gradient.

2.2.2 Aspect

Aspect was measured in azimuth degree in this study. Pictorial representation of azimuth degrees is shown in Figure 1. An azimuth is the direction measured in degrees clockwise from north on an azimuth circle (**https://www.nwcg.gov/course/ffm/location/62-azimuths**).

Two binary variables (north- yes/no & east- yes/no) are created to capture this information. For example azimuth degree 50 meaning north = 1 and east = 1, implies that unit of observation is in the upper right quadrant. The original variable 'Aspect' was dropped

Figure 1: Azimuth Degree



2.2.3 Soil Types

The 40 binary columns capturing soil type are collapsed into 11 general soil type columns (See Section 2.1). The original columns were dropped.

2.3 Spliting data into train and test sets

Cover type is the dependent variable in this analysis. This is an un-ordered qualitative variable with seven levels. The prevalence's of theses types in the full dataset is shown below.

Table 3: Prevalence of Cover Types

Cover_Type	Number	Proportion
Lodgepole Pine	283301	0.488
Spruce-Fir	211840	0.365
Ponderosa Pine	35754	0.062
Krummholz	20510	0.035
Douglas-fir	17367	0.030
Aspen	9493	0.016
Cottonwood/Willow	2747	0.005

This is a large dataset with close to 600,000 observations. The choice of proportion of data to be split as test set is based on the prevalence of the cover types and computation time.

Cottonwood/Willow type had the lowest prevalence of about 0.5 % (~ 2800 observations). The function **createDataPartition** sample within each class and will therefore ensure prevalence's of each type similar to

the prevalence's observed in the entire data. A sample of 10000 observations will have around 50 observations of Cottonwood and this is around 2% of data (p = 0.02).

This p will ensure that all tree cover types has some representation in the dataset, choosing cross-validation sets similar to test set size will also reduce computation time to some extent (especially KNN).

```
set.seed(27, sample.kind = "Rounding") # if using R 3.5 or earlier, use `set.seed(27)`
test_index<- createDataPartition(y = dat$Cover_Type, p = 0.02, list = F)
test_set<- dat[test_index, ]
train_set<- dat[- test_index, ]
remove(test_index, dat)
```

The number of observations in train and test set's are 569388 & 11624 respectively. Exploratory data analysis is done using only the train set.

2.4 Exploaratory Data Analysis

The association between the independent and dependent variables are presented in this section. Correlation between the independent variables are also explored.

2.4.1 Wildnerness area

This study was conducted in four wilderness area. The association between the cover types and these areas are tabulated below. The table displays proportion within each column (Cover Types).

Table 4: Cover Types by Wilderness Areas

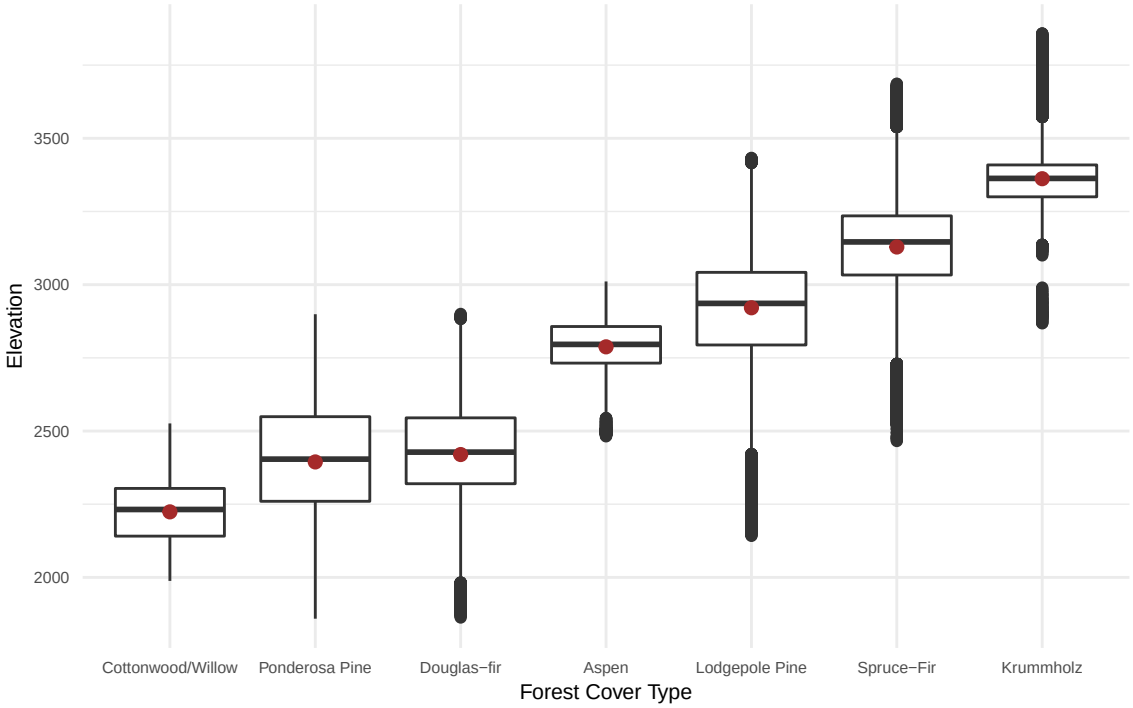
	Spruce-Fir	Lodgepole Pine	Ponderosa Pine	Cottonwood/Willow	Aspen	Douglas-fir	Krummholz
Cache la Poudre	0.000	0.011	0.6	1	0.000	0.559	0.000
Comanche Peak	0.413	0.442	0.4	0	0.602	0.441	0.640
Neota	0.088	0.032	0.0	0	0.000	0.000	0.112
Rawah	0.499	0.516	0.0	0	0.398	0.000	0.248

Comanche Peak appears to be the most diverse of the four areas. It has all tree types except Cottonwood, which is found only in Cache la Poudre. Lodgepole Pine is the only tree found in all four area.

2.4.2 Elevation

The distribution of elevation by forest cover types are shown in Figure 2. The dot inside the plot is the mean elevation. Elevation is markedly different between some of the cover types. Ponderosa Pine & Douglas-fir have very similar mean and distribution. Cottonwood & Krummholz have the lowest and highest mean elevation respectively.

Figure 2: Distribution of Elevation by Cover Types



2.4.3 Aspect

The original variable aspect in degree's was converted to two new variables (Section 2.2.2). The association between cover types and the four quadrants formed by these two variables are tabulated below.

Table 5: Cover Types by Aspect Quadrants

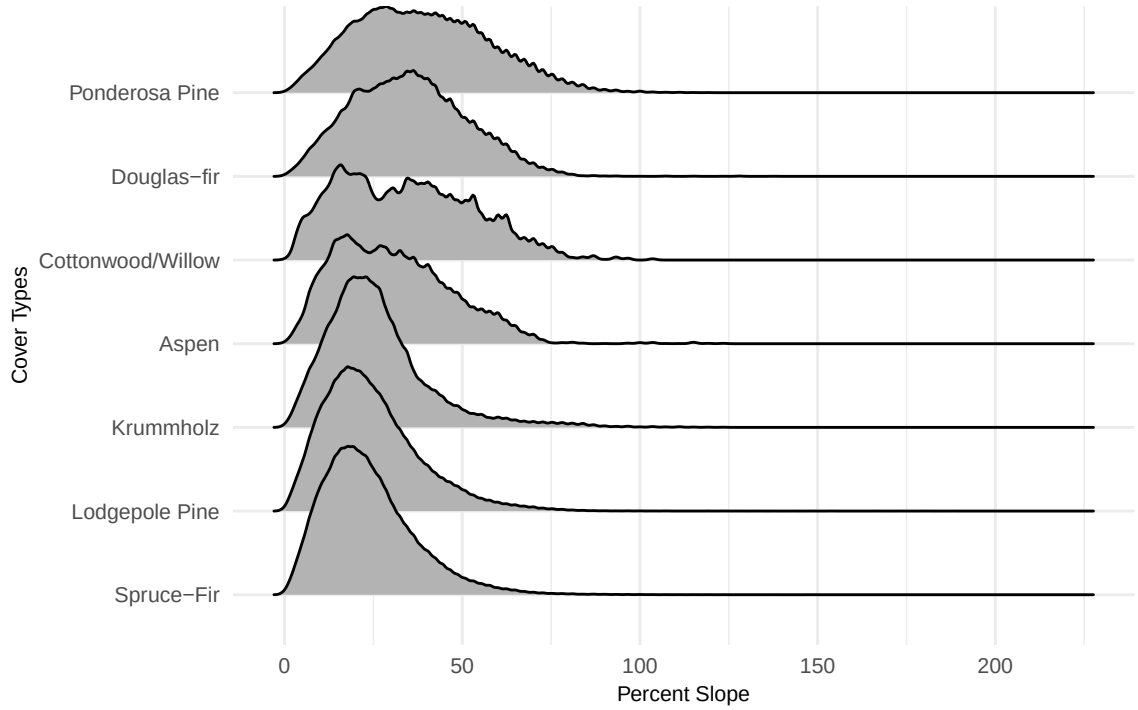
	Spruce-Fir	Lodgepole Pine	Ponderosa Pine	Cottonwood/Willow	Aspen	Douglas-fir	Krummholz
North & East	0.407	0.380	0.254	0.282	0.385	0.407	0.368
North & West	0.268	0.202	0.275	0.121	0.138	0.422	0.231
South & East	0.209	0.255	0.306	0.502	0.342	0.098	0.303
South & West	0.115	0.163	0.165	0.095	0.136	0.073	0.099

Douglas-fir is mostly observed in the upper quadrants - (0.40 & 0.42 in North East and North West respectively). Aspen and Cottonwood are mostly (>70%) seen on the Eastern side.

2.4.4 Slope

There is considerable overlap in the distribution of slope (Figure 3). Steeper slopes seems to be more common in areas were Aspen, Cottonwood, Douglas-fir and Ponnderosa Pine grow.

Figure 3: Distribution of Slope by Cover Types



2.4.5 Distance to Roadways and Fire-points

Mean and standard deviation (SD) of distance to roadway and fire ignition points are tabulated below.

Table 6: Distance to Roadways

	Spruce-Fir	Lodgepole Pine	Ponderosa Pine	Cottonwood/Willow	Aspen	Douglas-fir	Krummholz
Mean	2614.698	2429.191	943.963	913.464	1350.739	1038.406	2736.888
SD	1498.032	1618.372	614.685	365.954	1042.715	571.358	1201.404

Table 7: Distance to Fire Points

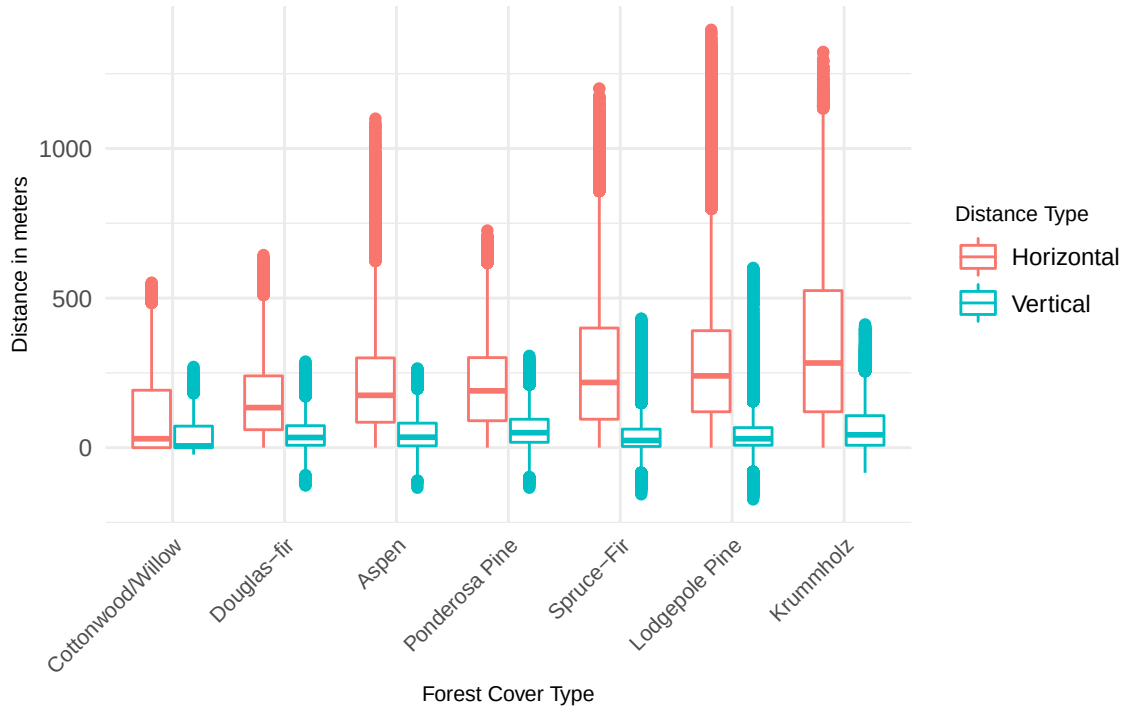
	Spruce-Fir	Lodgepole Pine	Ponderosa Pine	Cottonwood/Willow	Aspen	Douglas-fir	Krummholz
Mean	2009.577	2167.807	911.311	859.365	1580.314	1055.856	2068.556
SD	1234.608	1423.581	527.622	481.551	1000.348	577.018	1087.136

Lodgepole, Spruce-Fir and Krummholz appear farther away from roadways and fire ignition points when compared to others. There also appears to be correlation between the two variables when looked at summary level. The correlations between all the quantitative independent variables are presented in Section 2.4.8.

2.4.6 Distances to Hydrology

Both horizontal and vertical distance to the nearest water features were measured in meters. Negative values are possible for the vertical distances.

Figure 4: Distance to Hydrology by Cover Type



Except for Cottonwood which appears to be found very near the water features with overlap in horizontal and vertical distances, there are no pattern found on the other tree types.

2.4.7 Hillshade Index

Hillshade index (values from 0 to 255) was captured at 9 am, noon and 3 pm during summer solstice. Summer solstice is the longest sunlight period and occurs sometime between June 20 and June 22 in the Northern Hemisphere (**https://en.wikipedia.org/wiki/Summer_solstice**). Mean and SD of these variables are tabulated below.

Table 8: Hillshade Index 9 AM

	Spruce-Fir	Lodgepole Pine	Ponderosa Pine	Cottonwood/Willow	Aspen	Douglas-fir	Krummholz
Mean	212.021	213.843	201.928	228.324	223.476	192.82	216.968
SD	24.811	24.919	40.639	24.183	22.777	33.57	23.420

Table 9: Hillshade Index Noon

	Spruce-Fir	Lodgepole Pine	Ponderosa Pine	Cottonwood/Willow	Aspen	Douglas-fir	Krummholz
Mean	223.423	225.330	215.830	217.007	219.060	209.857	221.736
SD	18.160	18.508	27.917	20.864	24.908	24.427	20.033

Table 10: Hillshade Index 3 PM

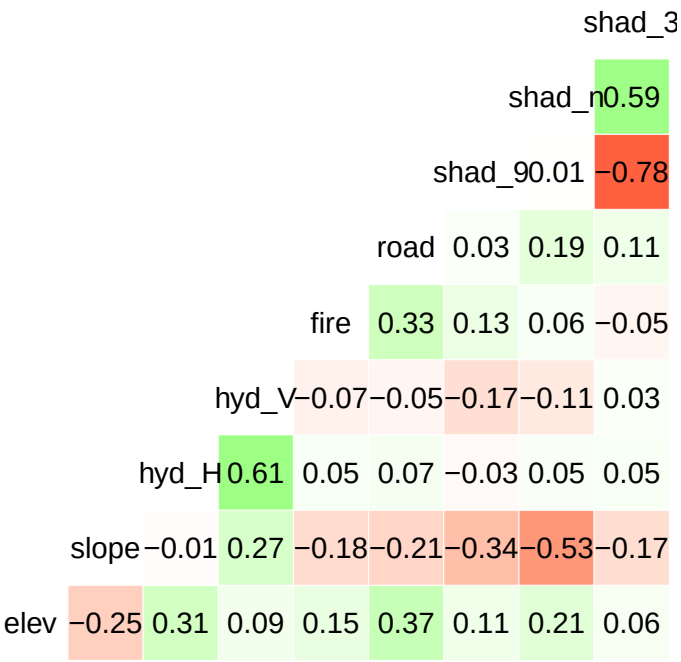
	Spruce-Fir	Lodgepole Pine	Ponderosa Pine	Cottonwood/Willow	Aspen	Douglas-fir	Krummholz
Mean	143.841	142.988	140.356	111.436	121.942	148.349	134.915
SD	36.048	36.227	52.499	49.218	49.447	45.380	38.945

Correlation is expected between these variables and other variables derived from the spatial data especially slope. This is because hillshading procedure uses slope and aspect in its computation.

2.4.8 Correlated variables

Correlation between all quantitative variables in the dataset are shown in figure below. There are nine variables as aspect was converted into two binary variables.

Figure 6: Correlation Matrix



The maximum absolute correlation was between hillshade index at 9 am and hillshade index at 3 pm (-0.78). The algorithm described by Kuhn and Johnston **(https://scientistcafe.com/ids/collinearity.html)**, offer an approach to decide which among the two highly correlated variable to be removed (variable with the higher average absolute correlation). The function *findCorrelation* of the caret package compute this average and returns the name of the variable. Hillshade index at 3 pm has a higher absolute correlation than the index at 9 am and this variable will not included in the prediction models.

```
findCorrelation(cor(cor_dat), cutoff = 0.75, names = T)
```

```
[1] "shad_3"
```

```
remove(cor_dat)
```

2.4.9 Cover Type & Soil Types

The proportion of observations of the 11 soil types (1 = present) are shown below. The function *nearZeroVar* is used to identify types with very small number of "present" compared to overall size.

```
train_set %>% select(17:27) %>% summarise_all(.funs = function(x) mean(x == 1))
```

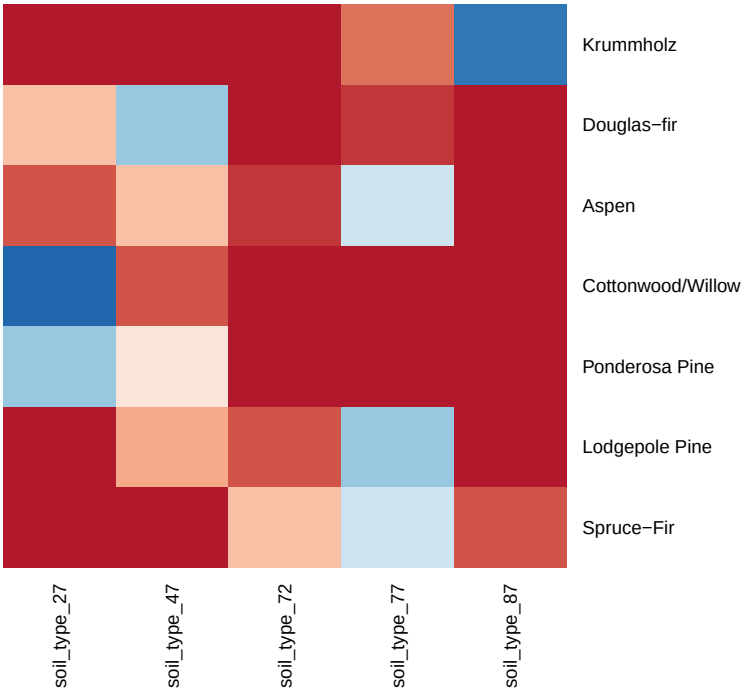
```
  soil_type_27 soil_type_35 soil_type_42 soil_type_47 soil_type_51 soil_type_61
1   0.06185764 0.0004882435  0.001986343    0.1590427  0.001043225   0.01078526
  soil_type_67 soil_type_71 soil_type_72 soil_type_77 soil_type_87
1  0.003254371   0.02433666    0.1568122    0.5107607   0.06963266
```

```
nearZeroVar(train_set %>% select(17:27), names = T)
```

```
[1] "soil_type_35" "soil_type_42" "soil_type_51" "soil_type_61" "soil_type_67"
[6] "soil_type_71"
```

Soil types 35, 42, 51, 61, 67 & 71 have proportion less than 0.05 and because these variables can take only two values (0 or 1), they will satisfy both the unique cut (< 10) and frequency cut ($> 95/5$) defaults of the function. These soil types will not be included in the prediction models. The relationship between the other soil types and tree cover types are shown in Figure 7.

Figure 7: Heatmap of Soil Types and Cover Types



Color represents the proportion of observations within cover type, blue meaning higher proportion and red lower. A high percentage of areas were Krummholz grow have soil type 87. Cottonwood mostly grows in soil type 27.

3. Results

The dataset donor in his article used overall **accuracy** as the performance metric to evaluate different techniques. The article presented two algorithms- Linear Discriminant Analysis (LDA) & Artificial Neural Network (ANN) using different number of predictors. The highest overall accuracy reported was 70.6 % and this achieved with ANN using all 54 predictors. The highest accuracy with LDA was 58.4 % [2].

The same metric (**Accuracy**) will be used in this analysis for comparison purpose. In this report, three machine learning techniques are used.

- Multinomial Logistic Regression
- K- Nearest Neighbor (k-NN)
- Classification Tree

The variables which were flagged in Section 2 will not be used in all three models and therefore removed from both train and test sets. A few R code chunks are printed in this section for ease of review.

```
train_set<- train_set %>%
  select(-c(8, 18:19, 21:24)) %>%  # 8 = hillshade_3, 18 = soil_35, 19 = soil_42, 21 to 24 are
                                   # soil_types 51,61, 67 and 71 respectively
  mutate(Cover_Type = factor(case_when(   # changing names to abbreviations to make
                                          # confusion matrix fit in a line
    Cover_Type == "Spruce-Fir" ~ "SF",
    Cover_Type == "Lodgepole Pine" ~ "LP",
    Cover_Type == "Ponderosa Pine" ~ "PP",
    Cover_Type == "Cottonwood/Willow" ~ "CW",
    Cover_Type == "Aspen" ~ "AS",
    Cover_Type == "Douglas-fir" ~ "DF",
    Cover_Type == "Krummholz" ~ "KR"
  )))

test_set<- test_set %>%            # same as train set
  select(-c(8, 18:19, 21:24)) %>%
  mutate(Cover_Type = factor(case_when(
    Cover_Type == "Spruce-Fir" ~ "SF",
    Cover_Type == "Lodgepole Pine" ~ "LP",
    Cover_Type == "Ponderosa Pine" ~ "PP",
    Cover_Type == "Cottonwood/Willow" ~ "CW",
    Cover_Type == "Aspen" ~ "AS",
    Cover_Type == "Douglas-fir" ~ "DF",
    Cover_Type == "Krummholz" ~ "KR"
  )))
names(test_set)
```

```
 [1] "Elevation"                         "Slope"
 [3] "Horizontal_Distance_To_Hydrology"  "Vertical_Distance_To_Hydrology"
 [5] "Horizontal_Distance_To_Roadways"   "Hillshade_9am"
 [7] "Hillshade_Noon"                    "Horizontal_Distance_To_Fire_Points"
 [9] "rawah"                             "neota"
[11] "comanche_peak"                     "cache_la_poudre"
[13] "Cover_Type"                        "north"
[15] "east"                              "soil_type_27"
[17] "soil_type_47"                      "soil_type_72"
[19] "soil_type_77"                      "soil_type_87"
```

3.1 Multinomial Logistic Regression

Multinomial logistic regression is the name for logistic regression where the dependent variable is nominal (un-ordered qualitative) with more than two levels. The ***multinom*** function of the ***nnet*** package is used in this analysis. Details on other functions that implements this regression can be found in this website **(https://www.analyticsvidhya.com/blog/2016/02/multinomial-ordinal-logistic-regression/)**.

The quantitative variables are scaled before fitting the model (train) and prediction (test). The variable "rawah" is removed form the data for this analysis. The four wilderness area was given as four binary variables and keeping all four would cause co-linearity with the intercept. The cover type is additionally removed in the test set. All independent variable printed above are included and there is no tuning parameter for this model.

The output type available with ***predict*** function for *multinom* fit is similar to other linear models discussed in the course [3]. It could be "prob" for probability which returns the probability for each class levels. The class with the highest probability is returned if type = "class" is given. Then confusion matrix is used to compute the performance metrics (See below).

```
scale2 <- function(x) (x - mean(x)) / sd(x) # the scale function in base returns object of
                                            # class = matrix; this custom function
                                            # preserves the vector class

multinom_train<- train_set %>%
  select(- rawah) %>%                              # dropping 'rawah' to avoid collinearity
  mutate_at(c(1:8), .funs = scale2)                # scaling the continuous variables

multinom_test<- test_set %>% select(- rawah, -Cover_Type) %>% # removing dependent variable
  mutate_at(c(1:8), .funs = scale2)

multinom_fit<- multinom(Cover_Type ~ ., data = multinom_train, trace = FALSE) # model fit

y_hat_mutlinom<- predict(multinom_fit, newdata = multinom_test, type = "class")

accuracy_multinom<- confusionMatrix(y_hat_mutlinom, test_set$Cover_Type)

accuracy_multinom$table
```

```
          Reference
Prediction   AS   CW   DF   KR   LP   PP   SF
        AS    0    0    0    0    1    0    0
        CW    0   10    1    0    0    7    5
        DF    2    5  102    0   35  105    1
        KR    0    0    0  220    2    0   80
        LP  182    0   68    5 4531   74 1168
        PP    5   40  177    5   71  530    5
        SF    1    0    0  181 1027    0 2978
```

```
accuracy_multinom$overall[c("Accuracy", "AccuracyLower", "AccuracyUpper")]
```

```
     Accuracy AccuracyLower AccuracyUpper
    0.7201480     0.7118895     0.7282961
```

```
remove(multinom_fit, multinom_test, multinom_train, y_hat_mutlinom, accuracy_multinom)
```

The predictions are generally poor in Aspen, Cottonwood and Douglas-fir, the three types with the three lowest prevalence (Table 3). Although Krummholz (3.5%), has a prevalence close to Douglas-Fir (3%), the algorithm detected this type better. The overall accuracy is 72 % and it is comparable to the highest accuracy reported by the dataset donor with ANN and much better than the one reported with LDA [2].

3.2 K- Nearest Neighbor (k-NN)

KNN is an algorithm that uses distance to predict observations. The k refers to the number of nearest neighbor to consider in making prediction and it is a tuning parameter.

For deciding the best k to use in test set, three cross-validation set of sample size similar to the test set are created. For this the function **createFolds** is used. This function splits the total observation in a set into k **(this k refers to k-fold CV; not to be confused with the k in KNN)** sets, preserving the balance of the class types (prevalence). A k of 50 will return a list of 50 index of size similar to the test size and the first three are extracted.

In each fold, KNN algorithm are run using k of 3, 7, 11 and 15. The creation of data folds and the KNN computations in the first fold are printed below. The quantitative variables are scaled and one of the wilderness area column are removed.

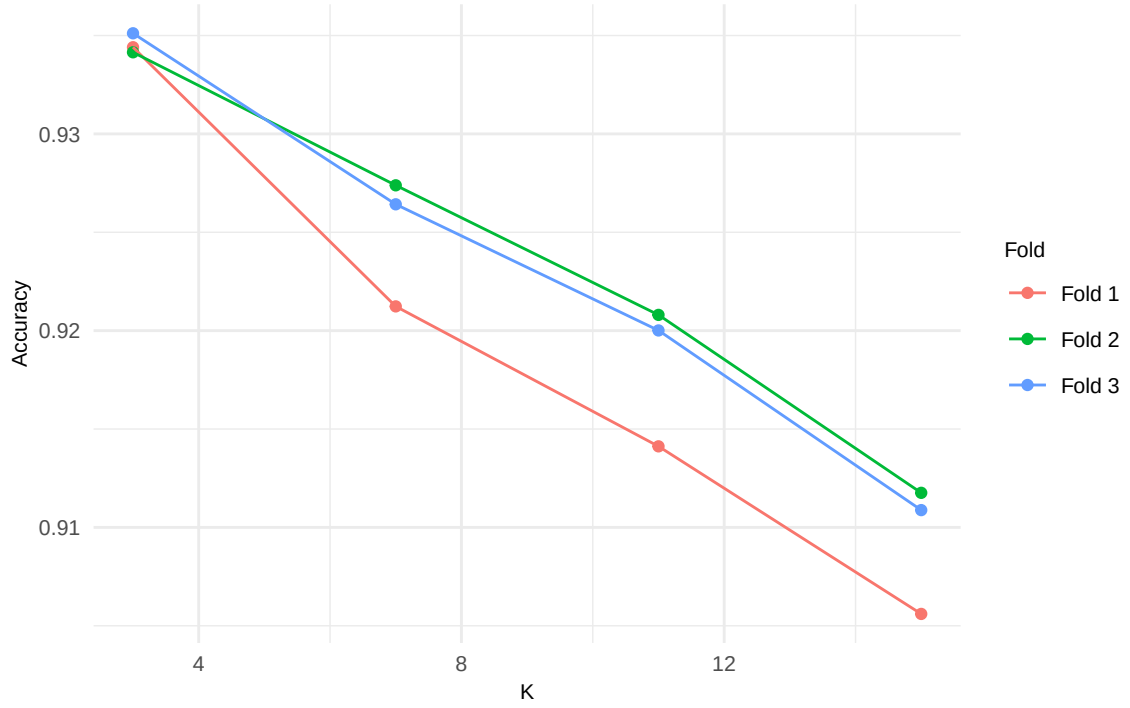
```
set.seed(27, sample.kind = "Rounding") # if using R 3.5 or earlier, use `set.seed(27)`

folds<- createFolds(y = train_set$Cover_Type, k = 50, list = T)[1:3] # indexing first 3 vectors

cv_fold1<- train_set %>% slice(folds$Fold01) %>%  # selecting fold 1 observations
  select(-Cover_Type, -rawah) %>%   # cover type removed - dependent, rawah removed
                                    # as it is redundant for wilderness information
  mutate_at(c(1:8), .funs = scale2) # scaling continuous variables (preprocessing for KNN)

x1<- train_set %>% select(- Cover_Type, - rawah) %>%  # same as above but for training
  slice(- folds$Fold01) %>%                           # removing fold 1 observations
  mutate_at(c(1:8), .funs = scale2) %>% as.matrix()

y1<- train_set$Cover_Type[- folds$Fold01]  # creating a vector of class

knn_fit1_k3<- knn3(x1, y1, k = 3)         # knn fit using k = 3

y_hat1_k3<- predict(knn_fit1_k3, cv_fold1, type = "class") # prediction in CV fold 1 set

f1_k3<- confusionMatrix(y_hat1_k3,
                        train_set$Cover_Type[folds$Fold01]) # computing performance metrics

remove(knn_fit1_k3, y_hat1_k3) # removing fitted model (k=3) and its predictions

knn_fit1_k7<- knn3(x1, y1, k = 7) # knn fit using k = 7
y_hat1_k7<- predict(knn_fit1_k7, cv_fold1, type = "class")
f1_k7<- confusionMatrix(y_hat1_k7, train_set$Cover_Type[folds$Fold01])
remove(knn_fit1_k7, y_hat1_k7)

knn_fit1_k11<- knn3(x1, y1, k = 11) # knn fit using k = 11
y_hat1_k11<- predict(knn_fit1_k11, cv_fold1, type = "class")
```

```
f1_k11<- confusionMatrix(y_hat1_k11, train_set$Cover_Type[folds$Fold01])
remove(knn_fit1_k11, y_hat1_k11)

knn_fit1_k15<- knn3(x1, y1, k = 15) # knn fit using k = 15
y_hat1_k15<- predict(knn_fit1_k15, cv_fold1, type = "class")
f1_k15<- confusionMatrix(y_hat1_k15, train_set$Cover_Type[folds$Fold01])
remove(knn_fit1_k15, y_hat1_k15, x1, y1, cv_fold1)

accuracy_fold1<- data.frame(Fold = "Fold 1", K = c(3,7,11, 15),
           Accuracy = c(f1_k3$overall["Accuracy"], f1_k7$overall["Accuracy"],
                        f1_k11$overall["Accuracy"], f1_k15$overall["Accuracy"]))
remove(f1_k3,f1_k7,f1_k11, f1_k15)
accuracy_fold1
```

```
    Fold  K  Accuracy
1 Fold 1  3 0.9344046
2 Fold 1  7 0.9212329
3 Fold 1 11 0.9141201
4 Fold 1 15 0.9056024
```

The above procedures were repeated for fold 2 and 3 and the results are plotted (Figure 8).

Figure 8: KNN Accuracy and K



Within all three folds, k = 3 produced the best accuracy and this value is applied for the test set. The result of the KNN algorithm for the test set is printed below.

```
x<- train_set %>% select(- Cover_Type, - rawah) %>%
  mutate_at(c(1:8), .funs = scale2) %>% as.matrix() # train set predictor matrix
```

```
y<- train_set$Cover_Type

knn_fit<- knn3(x, y, k = 3)  # best k from cross validation

knn_test<- test_set %>% select(- Cover_Type, - rawah) %>%
  mutate_at(c(1:8), .funs = scale2)   # test set data frame ; dependent variable removed

y_hat_knn<- predict(knn_fit, knn_test, type = "class")

accuracy_knn<- confusionMatrix(y_hat_knn, test_set$Cover_Type)

accuracy_knn$table
```

```
          Reference
Prediction   AS   CW   DF   KR   LP   PP   SF
        AS  145    0    2    0   22    4    5
        CW    0   39    2    0    0    9    0
        DF    0    4  290    0   10   38    0
        KR    0    0    0  399    1    0   18
        LP   40    1   13    1 5404   11  240
        PP    0   11   41    0   12  654    0
        SF    5    0    0   11  218    0 3974
```

```
accuracy_knn$overall[c("Accuracy", "AccuracyLower", "AccuracyUpper")]
```

```
     Accuracy AccuracyLower AccuracyUpper
    0.9381452     0.9336122     0.9424572
```

```
remove(x, y, knn_fit, knn_test, y_hat_knn, accuracy_knn)
```

There is a huge increase in accuracy from that obtained with Multinomial Logistic Regression. The performance of the algorithm in detecting cover types with low prevalence increases very much.

3.3 Classification Trees

Classification Tree is an algorithm that employs recursive (repeated) partitioning to fit data. The **rpart** function of the *rpart* package is used for this analysis. This algorithm has three tuning parameter - 'cp' (complexity parameter- proportion reduction in gini index or entropy), 'minsplit' (minimum number of observation in a node before splitting) and 'minbucket' (minimum number of observations in each node) which is minsplit/3 [3].

Considering that the prevalence of four of the seven tree covers are less than 0.05 (Table 3), with one type less than 0.005, the default tune parameter of cp = 0.01 and minsplit = 20 will not allow the tree grow past three or four nodes. To make this computation quicker minsplit was fixed at 8. **cp** of 0, 0.0025, 0.005 and 0.0075 was tested with cross validation sets.

CV procedure is similar to the one used with KNN. The wilderness information in four column are collapsed into one and the soil type are converted into factor for this algorithm. The method for first fold is shown below.

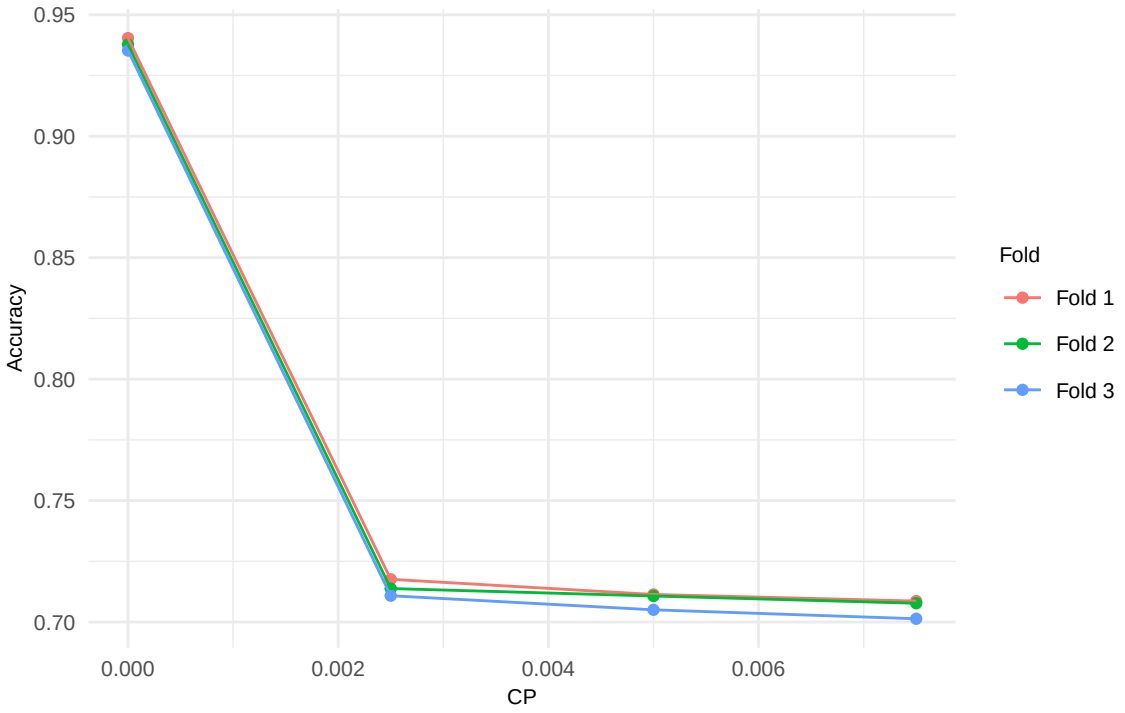
```r
set.seed(3, sample.kind = "Rounding") # if using R 3.5 or earlier, use `set.seed(3)`

folds<- createFolds(y = train_set$Cover_Type, k = 50, list = T)[1:3] # indexing first 3 vectors

rpart_train<- train_set %>%
  mutate_at(c(14:20), .funs = function(x)factor(ifelse(x == 1, "Yes", "No"))) %>%
  mutate(Wildnerness = factor(case_when(    # one column for wilderness information
    rawah == 1 ~ "Rawah",
    neota == 1 ~ "Neota",
    comanche_peak == 1 ~ "Comanche Peak",
    cache_la_poudre == 1 ~ "Cache la Poudre"
  ))) %>% select(- c(9:12))                # removing the binary wilderness columns


cv_fold1_train<- rpart_train %>%
  slice(- folds$Fold01)          # removing fold 1 observations

cv_fold1<-  rpart_train %>%
  slice(folds$Fold01) %>%       # slicing only fold 1 observation
  select(- Cover_Type)          # removing the variable to be predicted (dependent variable)

rpart_fit1_c1<- rpart(Cover_Type ~ ., data = cv_fold1_train, method = "class",
                  control = rpart.control(cp = 0, minsplit = 8))
y_hat_rpart1_c1<- predict(rpart_fit1_c1, cv_fold1, type = "class")
cm1<- confusionMatrix(y_hat_rpart1_c1, train_set$Cover_Type[folds$Fold01])
remove(rpart_fit1_c1, y_hat_rpart1_c1)

rpart_fit1_c2<- rpart(Cover_Type ~ ., data = cv_fold1_train, method = "class",
                  control = rpart.control(cp = 0.0025, minsplit = 8))
y_hat_rpart1_c2<- predict(rpart_fit1_c2, cv_fold1, type = "class")
cm2<- confusionMatrix(y_hat_rpart1_c2, train_set$Cover_Type[folds$Fold01])
remove(rpart_fit1_c2, y_hat_rpart1_c2)

rpart_fit1_c3<- rpart(Cover_Type ~ ., data = cv_fold1_train, method = "class",
                  control = rpart.control(cp = 0.005, minsplit = 8))
y_hat_rpart1_c3<- predict(rpart_fit1_c3, cv_fold1, type = "class")
cm3<- confusionMatrix(y_hat_rpart1_c3, train_set$Cover_Type[folds$Fold01])
remove(rpart_fit1_c3, y_hat_rpart1_c3)

rpart_fit1_c4<- rpart(Cover_Type ~ ., data = cv_fold1_train, method = "class",
                  control = rpart.control(cp = 0.0075, minsplit = 8))
y_hat_rpart1_c4<- predict(rpart_fit1_c4, cv_fold1, type = "class")
cm4<- confusionMatrix(y_hat_rpart1_c4, train_set$Cover_Type[folds$Fold01])
remove(rpart_fit1_c4, y_hat_rpart1_c4)

accuracy_fold1<- data.frame(Fold = "Fold 1",
                            CP = c(0, 0.0025, 0.005, 0.0075),
                            Accuracy = c(cm1$overall["Accuracy"],
                                         cm2$overall["Accuracy"],
                                         cm3$overall["Accuracy"],
                                         cm4$overall["Accuracy"]))

remove(cv_fold1, cv_fold1_train)
```

```
remove(cm1,cm2,cm3, cm4)
accuracy_fold1
```

```
    Fold     CP  Accuracy
1 Fold 1 0.0000 0.9403758
2 Fold 1 0.0025 0.7175975
3 Fold 1 0.0050 0.7113628
4 Fold 1 0.0075 0.7086407
```

The procedure above are repeated for fold 2 and fold 3 and the results are plotted in Figure 9.

Figure 9: Classification Accuracy and Tuning Paramters



With all three fold, highest accuracy was with cp = 0 which is almost similar to KNN. For all other values in all fold, the accuracy dropped significantly. There is considerable variation in accuracy across folds with cp of 0.0025, 0.005 and 0.0075, the accuracy with cp = 0.0025 in fold 3 was less than the accuracy obtained with cp = 0.005 in fold 1 and fold 2. Taking a conservative approach on not to over train, the cp of 0.0025 was chosen for the test set. The results are printed below.

```
rpart_test<- test_set %>%
  mutate_at(c(14:20), .funs = function(x)factor(ifelse(x == 1, "Yes", "No"))) %>%
  mutate(Wildnerness = factor(case_when(
    rawah == 1 ~ "Rawah",
    neota == 1 ~ "Neota",
    comanche_peak == 1 ~ "Comanche Peak",
    cache_la_poudre == 1 ~ "Cache la Poudre"
  ))) %>% select(- c(9:12)) %>% select(- Cover_Type) # removing the dependent variable

rpart_fit<- rpart(Cover_Type ~ ., data = rpart_train, method = "class",
                        control = rpart.control(cp = 0.0025, minsplit = 8)) # best cp
```

```
y_hat_rpart<- predict(rpart_fit, rpart_test, type = "class")

accuracy_rpart<- confusionMatrix(y_hat_rpart, test_set$Cover_Type)

accuracy_rpart$table
```

```
          Reference
Prediction   AS   CW   DF   KR   LP   PP   SF
        AS    0    0    0    0    0    0    0
        CW    0    0    0    0    0    0    0
        DF    0    0    0    0    0    0    0
        KR    0    0    0  199    3    0  102
        LP  181    0   76    3 4507   62 1174
        PP    8   55  272    0  131  654    1
        SF    1    0    0  209 1026    0 2960
```

```
accuracy_rpart$overall[c("Accuracy", "AccuracyLower", "AccuracyUpper")]
```

```
     Accuracy AccuracyLower AccuracyUpper
    0.7157605     0.7074645     0.7239483
```

```
remove(rpart_test, rpart_train, rpart_fit, y_hat_rpart, accuracy_rpart)
```

The accuracy obtained with this algorithm is comparable with the one obtained with Multinomial Logistic Regression and the accuracy reported by the dataset donor. The prediction pattern of tree types with low prevalence are similar to regression.

4. Conclusion

The cover type information are a part of forest management inventory and are useful for resources managers. The traditional method of collecting these information by field personnel or remote methods like aerial photography, using drones are costly and sometimes not possible- very large area, inaccessible or neighboring landmass [2].

These data are obtained from Landsat imaging and processing. Accuracy was chosen as the performance measure to make comparison with the result reported previously[2]. The accuracy obtained with Multinomial Logistic Regression was comparable to the one obtained with ANN (around 70%) by the data donor. The accuracy with LDA (57.8 %) was much lower than the ones reported here.

Given regression and linear discriminant analysis are somewhat related and produce similar results [3], this difference should be due to the sampling technique employed in that study. In that study, the training set was composed of 11340 observations with 1620 observations of each cover type [2].This is much lower than the number of observations used for training in this report. This is one of the strength of this analysis- a large training set along with a representative test set.

This analysis is not without limitations. Some are listed below.

- The most commonly used cut off absolute correlation (0.75) was used to remove highly correlated variables. For categorical variables, **nearZerovar** function was used with default. Other cut-offs for both could have completely different fit and these were not explored.

- Variable selection (backward, forward,.. in a separate CV set) can be considered as tuning in regression analysis and these were not performed in this analysis.
- Classification tree tuning parameters were not fully optimised- it is possible that a different cp and minsplit could have given a better accuracy
- Random Forest are powerful than classification tree and this technique was not used.
- Though not a big limiting factor, the number of cross validation was limited to three for computation quickness. Since the training set was very large- this would not have much impact.
- Aspect was measured in azimuth degree (Section 2.2.2). In this scenario, statistics with linear measurement will not apply. In an azimuth degree, 1 is closer to 350 than 120, as both 1 and 350 will lie in north side and 120 will be in the south. In this analysis, this was converted to binary variable. The use of circular statistics could have captured this information better.

References

1. FAO, Global Forest Resources Assessment 2000: Terms and definitions. Available at **http://www.fao.org/3/Y1997E/y1997e1m.htm#bm58**.

2. Blackard, Jock A. and Denis J. Dean. 2000. "Comparative Accuracies of Artificial Neural Networks and Discriminant Analysis in Predicting Forest Cover Types from Cartographic Variables." Computers and Electronics in Agriculture 24(3):131-151. Available at **https://www.fs.fed.us/rm/ogden/research/publications/downloads/journals/1999_compag_blackard.pdf**.

3. Rafael A. Irizarry. Introduction to Data Science. Available at **https://rafalab.github.io/dsbook/**