Analyzing and Predicting Forest Fires in the Montesinho Natural Park

Final project for Statistical Machine Learning, Summer A 2021

Group 5: Zachery Macintyre (zim2103) & Charlotte Wang (cw3204)

Abstract

In this project, we analyze a forest fires data set of the Montesinho Natural Park in Portugal. The data set documented 517 forest fires in the Montesinho Natural Park between January 2000 and August 2003. Using multiple linear regression, the lasso, stepwise selection models, and logistic regression, we analyzed possible models that predict forest fires. In addition, we used k-medoid clustering to explore different taxonomy of forest fires in the Montesinho Natural Park. The multiple linear regression analysis used to predict forest fire damage turned out not to be very informative because of the distribution of damage. Feature selection with lasso is useful for improving the model. In addition, we found that causes of small fires may not be systematically different from large fires. K-medoids clustering provides a possible taxonomy of the types of forest fires in the Montesinho Natural Park.

Introduction

Every year, forest fires worldwide damage personal property, cause air pollution, and create severe burdens to fire departments. It would be convenient if there were a way to predict fire damages more accurately. Such predictions would allow professionals to be more active in fire prevention and enable them to know how much staffing to send when a forest fire breaks out. In this project, we investigate to see if a model can accurately predict fire damage. Our primary goal is to create a model that has predictive power to forecast forest fires and estimate the potential damage. For example, the findings could show a few months are more prone to fires than others. On the other hand, if a fire starts, we can plug in some relevant variables to our models to find an appropriate fire response. Do you need your entire fire fighting team, or can the fire be quickly put out? Our analysis can expedite the process of responding to forest fires and prevent unnecessary danger for people in the area.

This project analyzed a data set that documents forest fire occurrences in Montesinho natural park, Portugal. Montesinho natural park is located in the northeast of Portugal and is a protected conservation area. The natural park features incredible biodiversity and attracts many tourists each year. In the next two sections, we lay out our research questions and introduce this data set.

Research Questions

Using this data set, we formulate the following research questions:

- Which features available here are more important for us to predict the scale of a forest fire? (Regression analysis)
- How can we use these features to predict a large fire? (Classification problem)
- Can we use these features to classify different types (causes) of forest fires? (Cluster analysis)

Data Overview

We acquired this data set from retriever.readthedocs.io. The data set can also be found at *http://archive.ics.uci.edu/ml/datasets/Forest+Fires*. To access the data set, we used the API calls from the retriever site and downloaded the .csv file to our folder for the final project.

According to [1], the data set was collected from January 2000 to August 2003. There are n = 517 observations (i.e., the total number of fires). For each observation, 13 features are available. These features are summarized in Table 1. There are five types of variables in this dataset. The first are location data (X and Y); the park has been separated into 81 grids in this data set. The second type of variables are temporal, qualitative variables (month and day). The third type comes from Canadian Forest Fire Weather Index (FFMC, DMC, DC, and ISI); these indices are widely used in the study of forest fires worldwide, and the measurements come from lagged moisture conditions. The fourth type of variables are real-time weather conditions (temp, RH, wind, and rain). The last variable is burned area (area).

The data set comes from two sources [1]. Fire occurrences and fire features, including location, time, FWI, and burned areas, were collected by inspectors of the Montesinho natural park. The Braganca Polytechnic Institute measured weather data. If the works were appropriately managed by the personnel from the two sources, we suspect that the validity of the data set should be satisfactory.

Table 1: Variable explanations

Variable name	Variable explanation
X	X-axis coordinates for spatial location (1-9) on a 9*9 grid
Y	Y-axis coordinates for spatial location (1-9) on a 9*9 grid
month	Month of the year (January - December)
day	Day of the week (Monday - Sunday)
FFMC	Fine Fuel Moisture Code (Canadian Forest Fire Weather Index, FWI): moisture of shaded litter fuels. Equivalent to 16-hour lagged moisture conditions.

DMC	Duff Moisture Code (FWI): moisture of shaded decomposed organic materials. Equivalent to 15-day lagged moisture conditions.
DC	Drought Code (FWI): drying deep in the soil. Equivalent to 53-day lagged moisture conditions.
ISI	Initial Spread Index (FWI): an index for spread potentials that combines moisture for fine dead fuels and wind speed.
temp	Temperature (degree Celsius)
RH	Relative humidity (%)
wind	Wind speed (km/hr)
rain	Accumulated precipitation within 30 mins (mm/m^2)
area	Total burned area (hectares)

Adapted from Table 1 in [1] and FWI information on

https://www.nwcg.gov/publications/pms437/cffdrs/fire-weather-index-system.

Much to our surprise and happiness, this data set has no missing values, so we did not have to make any calls to handle missing data. On top of that, we also did not have any values hidden within the data, so the data set is very clean and straightforward. Table 2 and 3 shows the summary statistics of this data set.

Table 2: Summary statistics for quantitative variables

	n	**mean**	**sd**	**min**	**max**	**range**	**se**
X	517	4.66924565	2.3137778	1	9	8	0.10175983
Y	517	4.29980658	1.2299004	2	9	7	0.05409096
FFMC	517	90.6446809	5.5201108	18.7	96.2	77.5	0.2427742
DMC	517	110.87234	64.0464822	1.1	291.3	290.2	2.81676107
DC	517	547.940039	248.066192	7.9	860.6	852.7	10.9099386
ISI	517	9.02166344	4.5594772	0	56.1	56.1	0.20052558
temp	517	18.8891683	5.8066253	2.2	33.3	31.1	0.25537509
RH	517	44.2882012	16.3174692	15	100	85	0.71764148
wind	517	4.01760155	1.7916526	0.4	9.4	9	0.07879679

rain	517	0.02166344	0.2959591	0	6.4	6.4	0.01301627
area	517	12.8472921	63.6558185	0	1090.84	1090.84	2.7995797

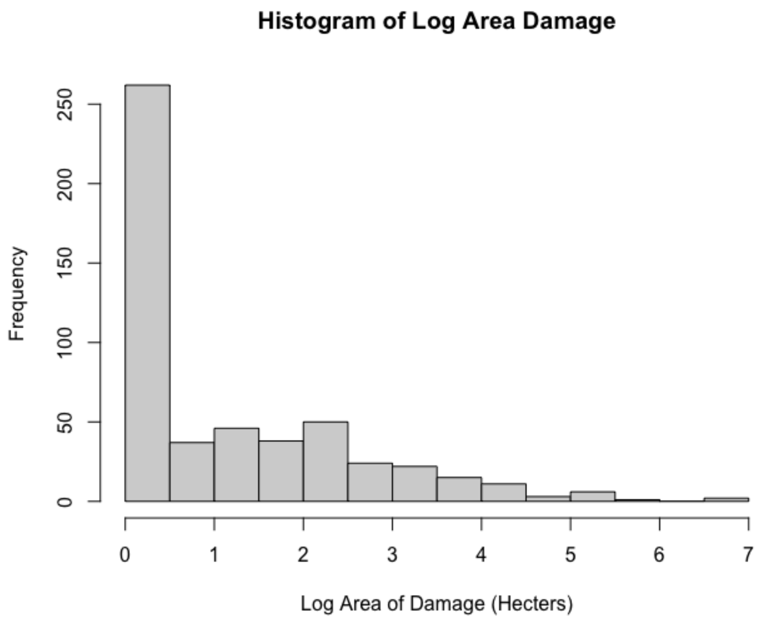
Table 3: Frequency table for qualitative variables

Month	**jan**	**feb**	**mar**	**apr**	**may**	**jun**	
	2	20	54	9	2	17	
	jul	**aug**	**sep**	**oct**	**nov**	**dec**	
	32	184	172	15	1	9	
Day	**mon**	**tue**	**wed**	**thu**	**fri**	**sat**	**sun**
	74	64	54	61	85	84	95

Data Analysis

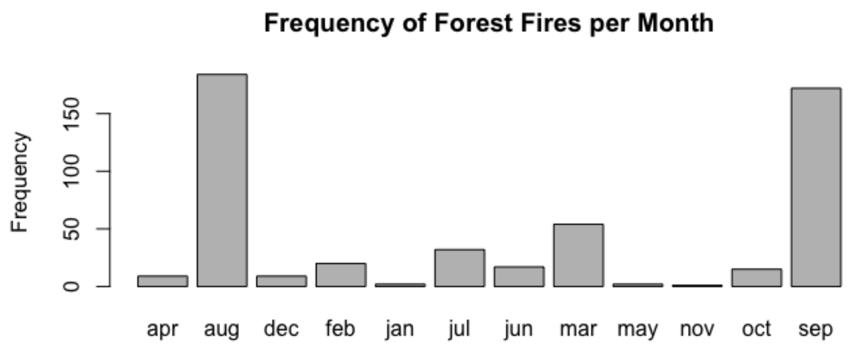
The first step in any machine learning problem is understanding what the data looks like. Since the burned area variable has a strong skew to the right, we use the log transformation log(area + 1), as recommended by [1], to adjust the data for better results. Graph 1 shows the overall distribution of the log(area+1) [“log area” hereafter] damaged in each fire. As we can see in this graph, most reported fires do very little damage. In fact, 247 out of 517 fires recorded in this data set shows a “0” burned area. According to [1], if a fire burns less than 0.01 hectares (= 100 m^2), a zero value will be recorded.

Graph 1: Histogram of log area damage



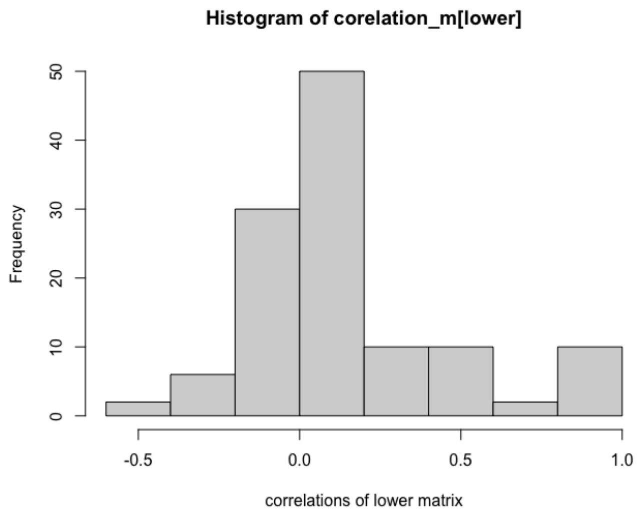
Graph 2 visualizes Table 2 and shows fires per month. We can see that there is a heavy concentration of fires during August and September. Such concentration intuitively makes sense, as these months are towards the end of the summer when the temperature is the highest and when the precipitation is the least. In other words, the forest is dry and hot, which is the perfect catalyst for forest fires.

Graph 2: Frequency of Forest Fires per month



Graph 3 shows the correlations between all the variables of interest. In this graph, we have used the lower correlation matrix to plot correlation values. We limit this chart to the lower matrix because the matrix is symmetric. The graph shows us that most variables are not correlated with each other, but around 15 highly correlated pairs of variables are correlated. Such correlation could potentially cause problems down the line. Additionally, it is essential to note that the values that take on 1 are just the variable paired with itself; with 10 variables, we have 10 correlation values of 1.

Graph 3: Histogram of lower correlation matrix



Regression Models

Can we predict fire damage accurately and gather information on which values are most important? To answer our first research question, we tried out many different multiple linear regression models. We started by splitting the data into a test set and a training set, and used 80% of the data for training and 20% for testing. We then used 10-fold cross validation on the training data. We looked at ten different multiple linear regression models. In our analysis, we will focus on overall trends and discuss only a few specific models. Table 4 shows the ten models. For all models, we used log_area = log(area + 1) as the response variable. In Model 1, we used all variables except for month, day, and location information (X and Y) as features. In Model 2, we used all variables excluding month and day as features. In Model 3, we conducted one-hot encoding to transform qualitative variables, month and day, and used all variables (excluding one dummy for each of the qualitative variables) to fit the linear model. In Models 4-6, we conducted forward stepwise selection, backward stepwise selection, and sequential selection on all variables. In Model 7, we conducted the lasso for a model with all variables. In Model 8, we fit a multiple linear regression but recoded the "day" variable as a binary weekday/weekend variable. In Model 9, we recoded the "month" variable as four-season dummy variables. In Model 10, we used the lasso on all variables with recoded day and month.

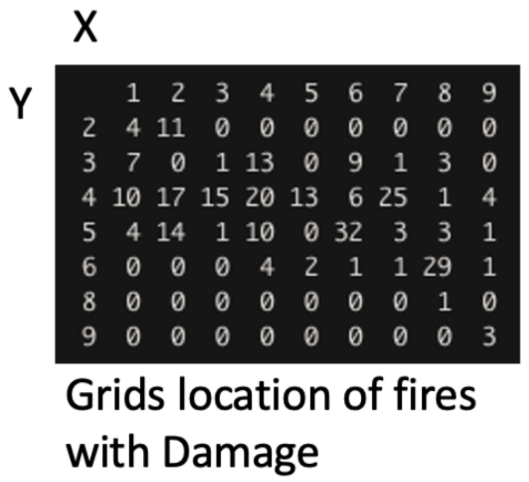
The regression analysis appears to do pretty well at first glance, as the highest test MSE in Table 4 is only 2.127. However, log damage area only takes on values in [0,6.99], making the test MSE not as impressive. The models we tested vary from an average of 30% off the true values (Model 3) to just being 12% off (Model 7). The 12% model (Model 7) seems not terrible, but upon further inspection, we find that the only factor showing significance in our training model is whether or not the month is December (month_dec). This feature is binary and only takes on a value of 1 or 0. So, essentially this model is a flat line, y = 1.106 + 0.287*(month is December). Even worse is that there are only 9 cases of fires in December, meaning only one percent of the fires occurred in that month, and the model almost uniformly estimates the damage to be 1.106 log hectors. This does not give us any important information on predicting fires or insights into factors affecting fires. The second-best performing model, Model 10, does just slightly worse than the Model 7. However, the problem here is that the coefficients for the important variables are microscopic, and two of them are binary variables; y = 1.110 + 0.00006*(wind) - 0.001*(season is spring) + 0.0002*(season is fall). We get pretty much the same result if we just know whether the month is December or not. We find that even after correction, the data set has an uneven distribution with fire damage; too many fires in our data set only caused negligible damages and were therefore coded as zero damage. However, since all observations in our data set recorded fire occurrences, we do not think it is reasonable to simply remove these observations either.

Table 4: Regression Model Outputs

Model	**Model 1: w/o month, day, and location variables**	**Model 2: w/o month and day variables**	**Model 3: All variables included**	**Model 4: Forward stepwise selection for all variables**	**Model 5: Backward stepwise selection for all variables**
Test MSE	1.578	1.574	2.217	1.522	1.499
Significance (important variables)	None	wind	month_aug, DMC, temp, wind	month_dec, temp	month_dec
Model	**Model 6: Sequential selection for all variables**	**Model 7: Lasso with all variables**	**Model 8: Recoding days to weekend/weekday**	**Model 9: Recoding months to seasons**	**Model 10: Lasso using weekend/ weekdays & seasons**
Test MSE	1.522	0.886	1.567	1.536	0.888
Significance (important variables)	month_dec, temp	month_dec	wind	wind, summer, temp	wind, spring, fall

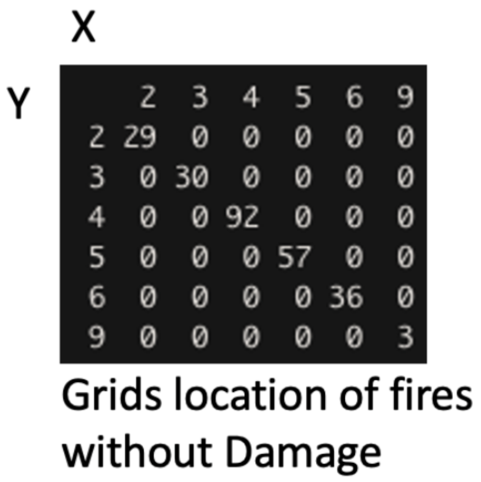
Interestingly, we observed a pattern in the grid positions of fires. Looking at Figure 1 below, we can see that there is a diagonal pattern of fires that cause zero damage (i.e. area=0). Figure 1 one shows that although diagonal grids seem to be more prone to fires, these fires tend to cause very little damage. On the contrary, large fires happened more often in rows 4 and 5. Compared to the map of the park (Figure 2), our best guess is that although inland grids (larger Y and smaller X; the southeast direction) are more prone to fires, due to the shape of the park, there might be more firefighters staffed in the diagonal grids. Therefore, fires in these grids were put out much more quickly and caused less damage. However, this is simply our rough guess and we have no knowledge on the staffing distribution in the park. We would need more domain knowledge to understand such a pattern.

Figure 1: Grid Locations of Fires



Y \ X	1	2	3	4	5	6	7	8	9
2	4	11	0	0	0	0	0	0	0
3	7	0	1	13	0	9	1	3	0
4	10	17	15	20	13	6	25	1	4
5	4	14	1	10	0	32	3	3	1
6	0	0	0	4	2	1	1	29	1
8	0	0	0	0	0	0	0	1	0
9	0	0	0	0	0	0	0	0	3

Grids location of fires with Damage



Y \ X	2	3	4	5	6	9
2	29	0	0	0	0	0
3	0	30	0	0	0	0
4	0	0	92	0	0	0
5	0	0	0	57	0	0
6	0	0	0	0	36	0
9	0	0	0	0	0	3

Grids location of fires without Damage

Figure 2: Map of Montesinho Natural Park

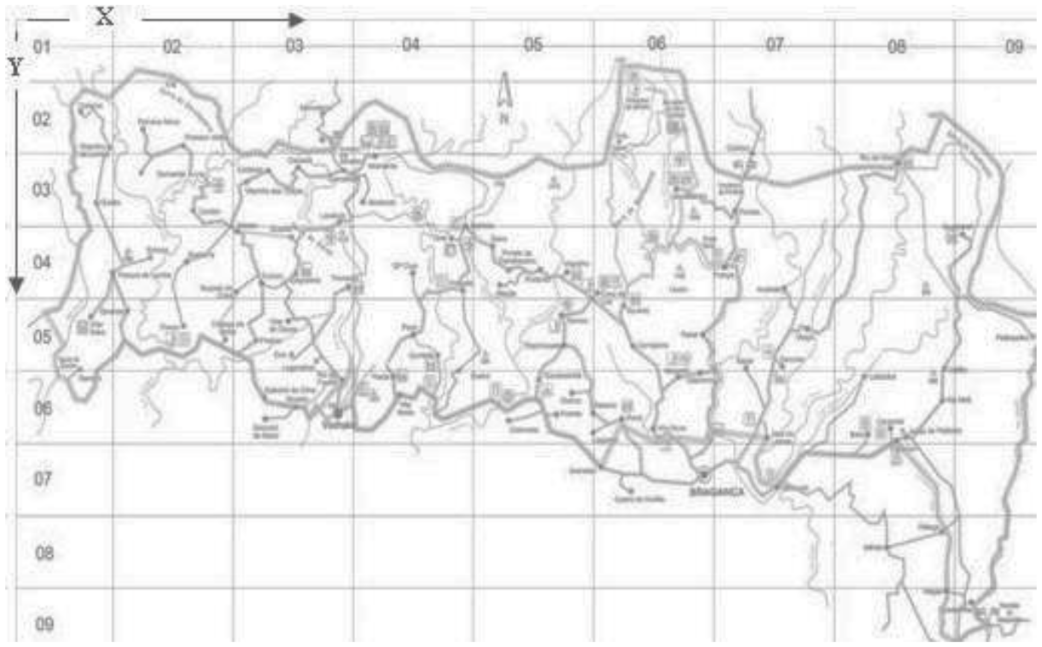


Figure from [1] Fig. 2

Classification Analysis

In this section, we dive deeper to further analyze the difference between large fires and small fires. Instead of using burned areas as the response variable, we separate all observations into two categories: large fires and small fires. We use the median value of log area, 0.42, as the cutoff between small and large fires. We create a binary variable, large_fire, which equals 1 if log area is greater than 0.42, and 0 otherwise. We then conduct logistic regression to build a model to classify small and large fires. Again, we used 80% of the data as the training set and 20% as the test set, and used 10-fold cross validation on the training data. We then predict probabilities and use the naive Bayes approach to construct confusion matrices. It turns out that contrary to our initial guess, large fires and small fires may not be intrinsically different. After trying four different logistic models, we see no significant variables in our training models. For our test results, according to confusion matrices, it turned out three models (Models 11-13) are only slightly better than randomly guessing, and the one after recoding (Model 14) is even worse than random guess. Results in Table 5 indicate that logistic models might not be suitable for our data.

Alternatively, we can use other classification methods, such as support vector machines or random forest to classify our data. Due to the limitation of time, we did not conduct these alternative methods. Another possible way to remedy our models is to adjust the cutoff between large and small fires. However, as mentioned previously, since 47.8% of observations in this data set have zero-value burned areas, we find our choice to use the median value of log area to separate large and small fires pretty reasonable. Under such labeling, almost all fires labeled as "small fires" have zero burned area. We therefore concluded that causes of small and large fires might not be intrinsically different, which explains why our logistic classification models did not work well.

Table 5: Logistic model outputs

Model	**Model 11: w/o month, day & location**	**Model 12: w/o month, day**	**Model 13: All variables**	**Model 14: Weekend/Weekday split & Seasonal split**
Test error rate	0.453	0.483	0.456	0.551
Test false positive rate	0.444	0.479	0.45	0.547
Test false negative rate	0.459	0.487	0.462	0.555
Better than random guessing?	Yes	Yes	Yes	No
Significant variables	None	None	None	None

Clustering Analysis

Finally, can we use variables in this data set to classify different types (causes) of forest fires? To answer this question, we conducted k-medoids clustering on our data set. It turns out that k-medoids might be useful for us to find patterns in this data set. Since clustering is an unsupervised method, for the following analysis, we no longer separate our data into training and test sets, but instead use all observations. To explore our data to the full potential, we use all variables in the dataset, including qualitative variables (after one-hot encoding).

When using k-medoids clustering, there are typically two ways to decide how many clusters to use. The first one is to plot the total within-sum-of-squares over the number of clusters, and look at where the "elbow" locates. Since the total within-sum-of-squares increase as the number of clusters increases, we use the number of clusters that result in low enough within-sum-of-squares without making the model too complicated. In Figure 3, we can see that the elbow is located at approximately k = 3. Another approach for finding the optimal number of clusters is to plot the gap statistics over the number of clusters (Figure 4) and use the number of clusters that results in the highest gap statistics. However, in this case, we find that the highest gap statistics can only be obtained with at least 15 clusters. Given that there are only 517 observations, k = 15 is likely to make our model overly complicated. Therefore, we decided to use three clusters in our k-medoids clustering. Figure 5 is a visualization of all observations in the dimensions of the first two principal components. It turns out the two principal components explain 12.7% and 6.5% of variances respectively, which is not perfect.

Next, we added our clustering analysis results to our original data set and see if these clusters help us better classify our observations. First, we fit a multiple linear model to verify the relationship between log_area and the three clusters. The regression result in Figure 6 shows that these clusters are indeed significantly correlated with different values of log burned areas. We further analyzed these relations by fitting one multiple linear model for observations in each cluster. The results are shown in Table 6; plus (+) and minus (-) signs after each variable indicates the relationship between the variable and log area. To interpret the results in Table 6, we can think of each cluster as a type of forest fire. For forest fires that belong to the first cluster, burned areas are most correlated with location. For forest fires in the second clusters, burned areas are mostly correlated with whether the fires happen in August and the level of the DMC index. Fires in the third cluster are more complicated; in this cluster, burned areas are correlated with the X-variable of location information, whether the month is August or is December, levels of DMC and DC indices, and the real-time relative humidity.

Due to the fact that clustering methods are unsupervised methods, there is no way to verify if our taxonomy here is feasible. However, we believe that such exploratory data analysis with clustering methods can provide more insights to our supervised data analysis if we have more observations and more features. We have also conducted hierarchical clustering methods, but the classification results did not show strong correlations with the log area variable after combining with the original data set. Due to the limitation of space, we decide to omit the results.

Figure 3: Total within-sum-of-squares and number of clusters

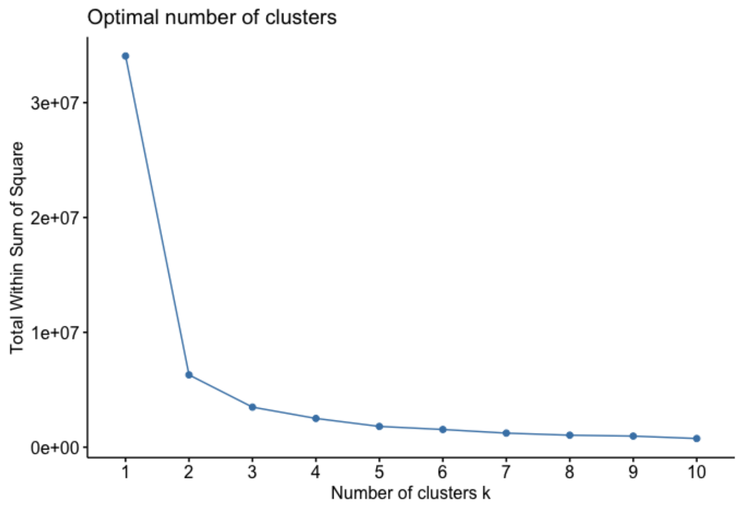


Figure 4: Gap statistics and number of clusters

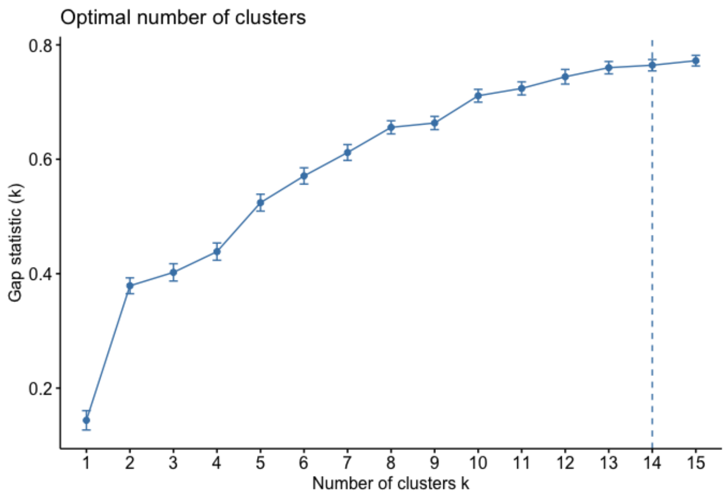


Figure 5: Cluster plot of the first two principal components

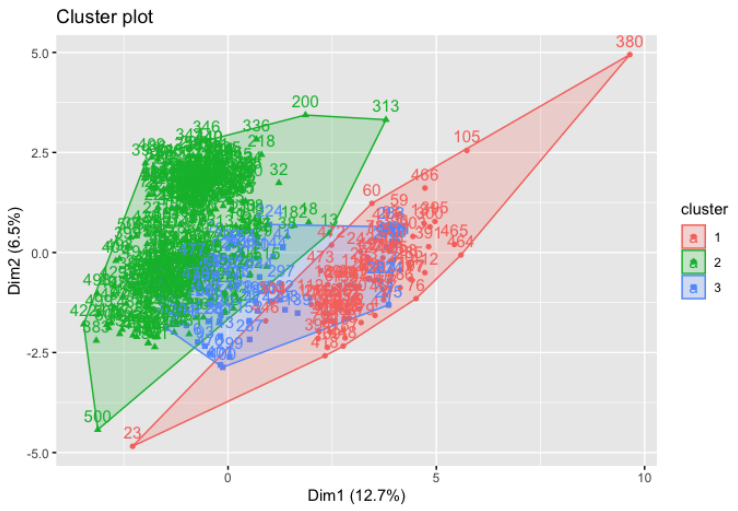
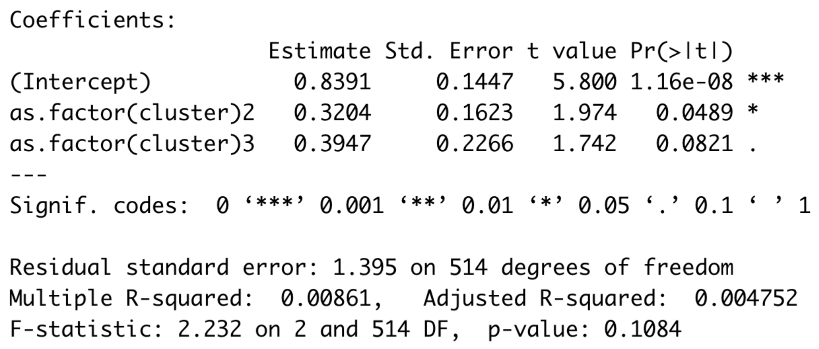


Figure 6: Relationship between log_area and the three clusters



```
Coefficients:
                    Estimate Std. Error t value Pr(>|t|)    
(Intercept)           0.8391     0.1447   5.800 1.16e-08 ***
as.factor(cluster)2   0.3204     0.1623   1.974   0.0489 *  
as.factor(cluster)3   0.3947     0.2266   1.742   0.0821 .  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.395 on 514 degrees of freedom
Multiple R-squared:  0.00861,   Adjusted R-squared:  0.004752 
F-statistic: 2.232 on 2 and 514 DF,  p-value: 0.1084
```

Table 6: Multiple linear regression results for each cluster

Cluster	**Cluster 1**	**Cluster 2**	**Cluster 3**
Significant variables	X (+), Y (-),	Month_aug (-), DMC(+)	X(+), month_aug(+), month_dec(+), DMC(+), DC(-), RH(+)

Conclusion

In this project, we analyze the forest fires data set of the Montesinho Natural Park. We fit multiple linear regression models that predict burned areas in the data set. We tried to find models that explain the difference between large fires and small fires. We also used clustering methods to explore different taxonomy of forest fires in the Montesinho Natural Park. The multiple linear regression models turned out not to be very informative because of the distribution of damage. Feature selection with lasso is useful for improving the model. We found that causes of small fires may not be systematically different from large fires. However, k-medoids clustering provides a possible taxonomy of the types of forest fires in the Montesinho Natural Park and shed light to more in-depth analysis of this data set.

If we have the resources to collect more complete data, we would like to gather better temporal information. For now, the data set only records months and days of fire occurrences, and other information such as years and dates are nonexistent. We were therefore unable to conduct any time-series analysis. In addition, we would also like to see data for days and areas in which fire did not occur. Obtaining this information can help us create a better classifier that can not only predict the scale of a fire but also whether a fire would break out at all.

Appendix

Reference

[1] P. Cortez and A. Morais. “A Data Mining Approach to Predict Forest Fires using Meteorological Data.” In J. Neves, M. F. Santos and J. Machado Eds., New Trends in Artificial Intelligence, Proceedings of the 13th EPIA 2007 - Portuguese Conference on Artificial Intelligence, December, Guimarães, Portugal, pp. 512-523, 2007. APPIA, ISBN-13 978-989-95618-0-9.

Additional Resources

http://www.sthda.com/english/articles/37-model-selection-essentials-in-r/154-stepwise-regression-essentials-in-r/
https://www.statology.org/k-medoids-in-r/
https://www.datacamp.com/community/tutorials/hierarchical-clustering-R

Workload distribution

Zac: Set up Github; coded initial data visualization, multiple linear regression, cross-validation, and stepwise selection; analyzed all linear regression results; wrote regression model and introduction sections
Charlotte: Coded summary statistics, lasso, logistic regression, k-medoids, and hierarchical clustering; wrote classification, clustering, conclusion, and abstract sections; edited the final write-up

Codes

See the next page

Analyzing and Predicting Forest Fires in the Montesinho Natural Park

Group 5: Zac Macintyre and Charlotte Wang

6/21/2021

```r
library(boot)
library(mltools)
library(data.table)
library(psych)
```

```
##
## Attaching package: 'psych'

## The following object is masked from 'package:boot':
##
##     logit
```

```r
library(glmnet)
```

```
## Loading required package: Matrix

## Loaded glmnet 4.0-2
```

```r
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x ggplot2::%+%()      masks psych::%+%()
## x ggplot2::alpha()    masks psych::alpha()
## x dplyr::between()    masks data.table::between()
## x tidyr::expand()     masks Matrix::expand()
## x dplyr::filter()     masks stats::filter()
## x dplyr::first()      masks data.table::first()
## x dplyr::lag()        masks stats::lag()
## x dplyr::last()       masks data.table::last()
## x tidyr::pack()       masks Matrix::pack()
## x tidyr::replace_na() masks mltools::replace_na()
## x purrr::transpose()  masks data.table::transpose()
## x tidyr::unpack()     masks Matrix::unpack()
```

```r
library(caret)
```

```
## Loading required package: lattice

##
## Attaching package: 'lattice'

## The following object is masked from 'package:boot':
##
##     melanoma

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```r
library(leaps)
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```r
library(cluster)
fires = read.csv("ForestFires.csv")
head(fires)
```

```
##   X Y month day FFMC  DMC    DC  ISI temp RH wind rain area
## 1 7 5   mar fri 86.2 26.2  94.3  5.1  8.2 51  6.7  0.0    0
## 2 7 4   oct tue 90.6 35.4 669.1  6.7 18.0 33  0.9  0.0    0
## 3 7 4   oct sat 90.6 43.7 686.9  6.7 14.6 33  1.3  0.0    0
## 4 8 6   mar fri 91.7 33.3  77.5  9.0  8.3 97  4.0  0.2    0
## 5 8 6   mar sun 89.3 51.3 102.2  9.6 11.4 99  1.8  0.0    0
## 6 8 6   aug sun 92.3 85.3 488.0 14.7 22.2 29  5.4  0.0    0
```

```r
colSums(is.na(fires))
```

```
##     X     Y month   day  FFMC   DMC    DC   ISI  temp    RH  wind  rain  area
##     0     0     0     0     0     0     0     0     0     0     0     0     0
```

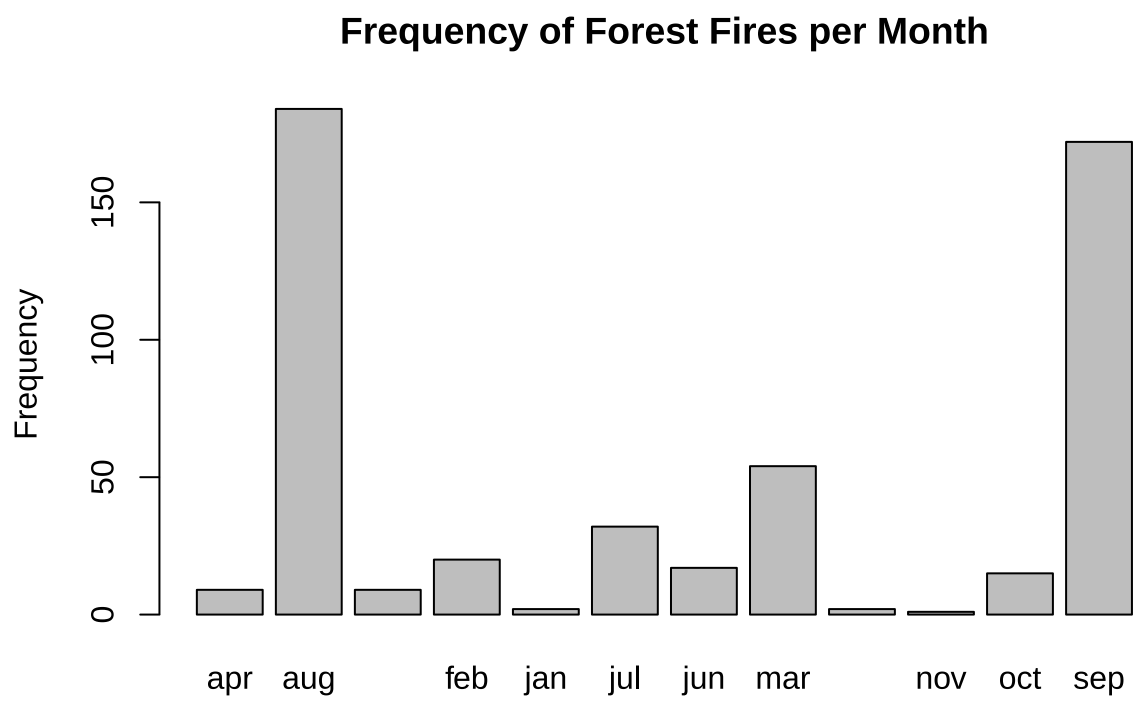
As we said in the presentation there is no missing values just doing a little check

This section is some initial data visualization and some data analysis

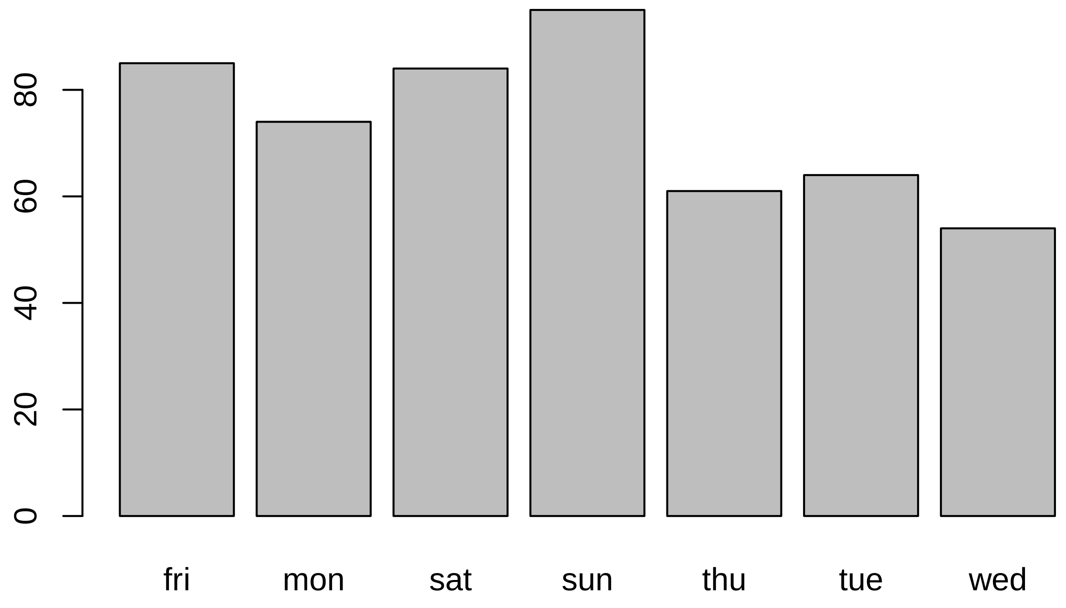
As suggested in Cortez and Morais (2007), transforming area to log(area+1) can correct area's skewness toward zero

```
log_fires = fires
log_fires$log_area = log(log_fires$area +1)
log_fires = log_fires[, -c(13)]
```

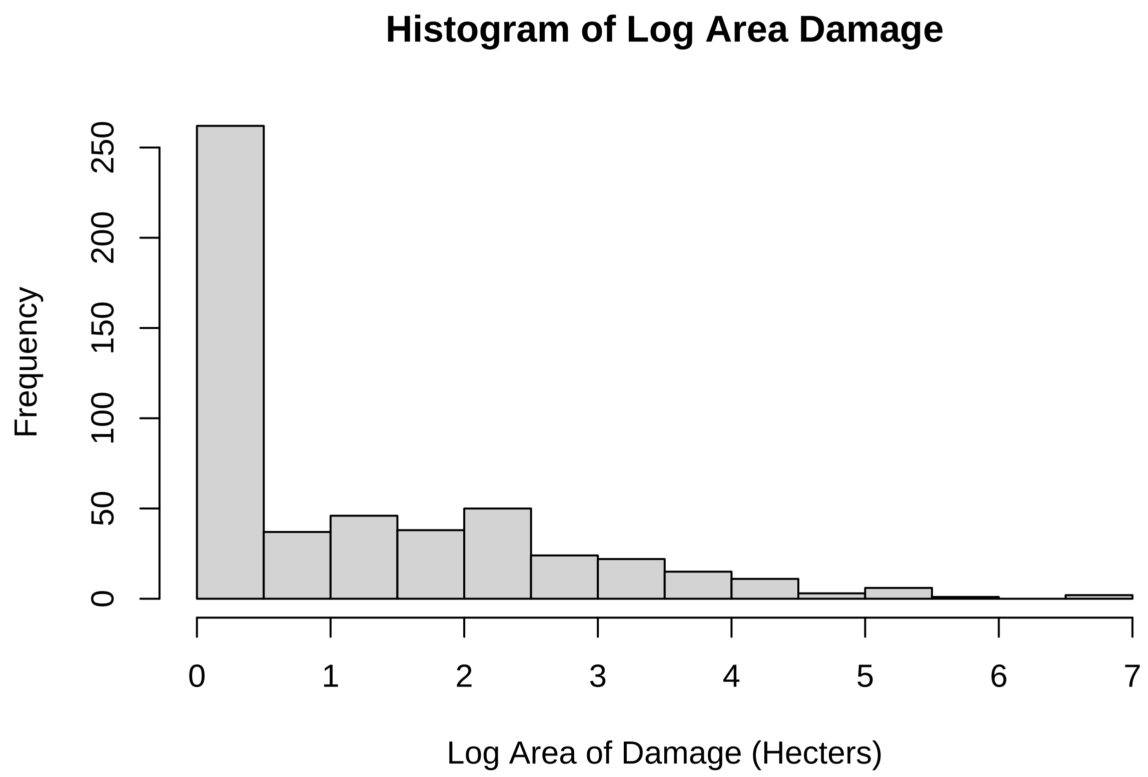
```
barplot(table(fires$month), ylab = "Frequency", main = "Frequency of Forest Fires per Month")
```



```
barplot(table(fires$day))
```



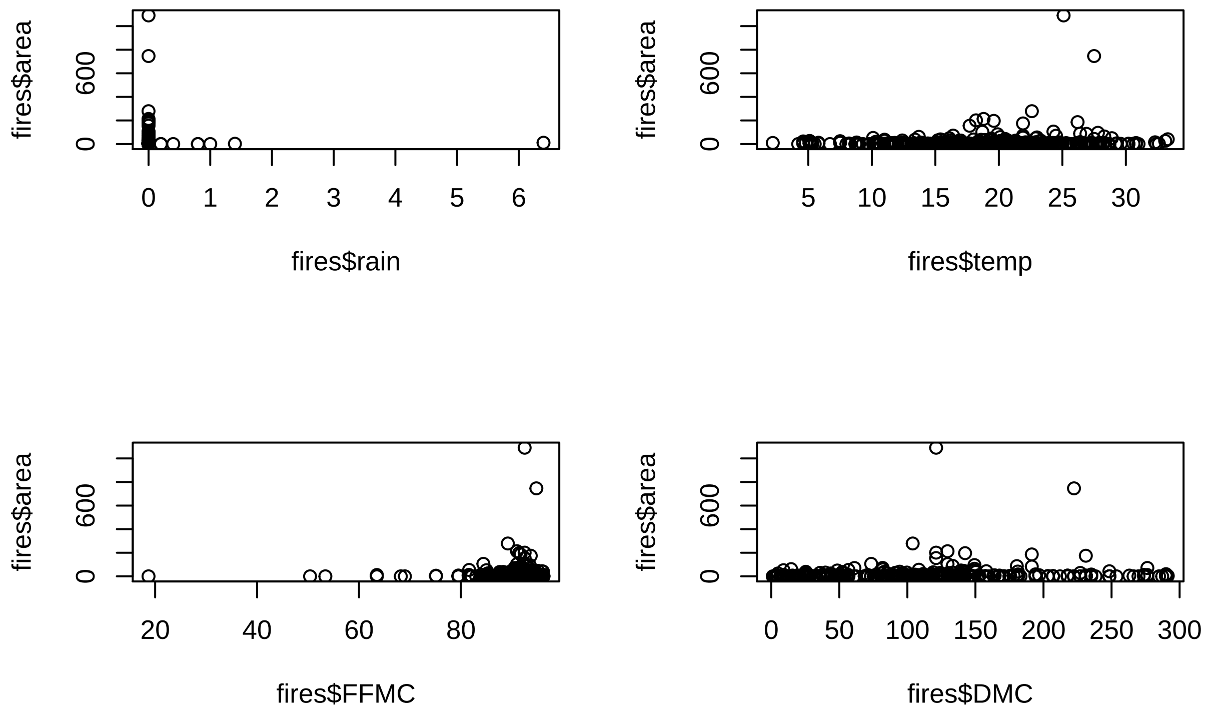
```r
hist(log_fires$log_area, xlab = "Log Area of Damage (Hecters)", main = "Histogram of Log Area Damage")
```



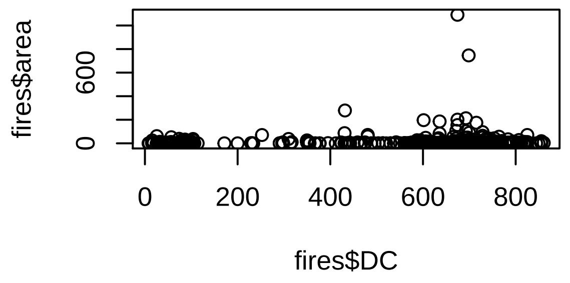
```r
summary(fires)
```

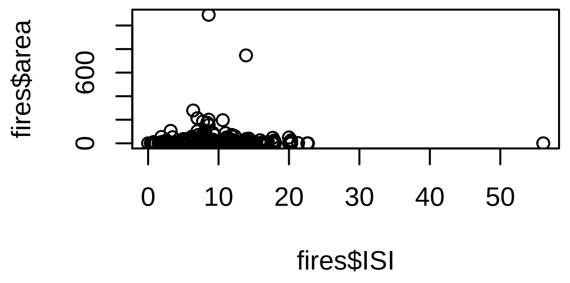
```
##        X               Y          month               day
##  Min.   :1.000   Min.   :2.0   Length:517         Length:517
##  1st Qu.:3.000   1st Qu.:4.0   Class :character   Class :character
##  Median :4.000   Median :4.0   Mode  :character   Mode  :character
##  Mean   :4.669   Mean   :4.3
##  3rd Qu.:7.000   3rd Qu.:5.0
##  Max.   :9.000   Max.   :9.0
##       FFMC            DMC              DC             ISI
##  Min.   :18.70   Min.   :  1.1   Min.   :  7.9   Min.   : 0.000
##  1st Qu.:90.20   1st Qu.: 68.6   1st Qu.:437.7   1st Qu.: 6.500
##  Median :91.60   Median :108.3   Median :664.2   Median : 8.400
##  Mean   :90.64   Mean   :110.9   Mean   :547.9   Mean   : 9.022
##  3rd Qu.:92.90   3rd Qu.:142.4   3rd Qu.:713.9   3rd Qu.:10.800
##  Max.   :96.20   Max.   :291.3   Max.   :860.6   Max.   :56.100
##       temp             RH              wind            rain
##  Min.   : 2.20   Min.   : 15.00   Min.   :0.400   Min.   :0.00000
##  1st Qu.:15.50   1st Qu.: 33.00   1st Qu.:2.700   1st Qu.:0.00000
##  Median :19.30   Median : 42.00   Median :4.000   Median :0.00000
##  Mean   :18.89   Mean   : 44.29   Mean   :4.018   Mean   :0.02166
##  3rd Qu.:22.80   3rd Qu.: 53.00   3rd Qu.:4.900   3rd Qu.:0.00000
##  Max.   :33.30   Max.   :100.00   Max.   :9.400   Max.   :6.40000
##       area
##  Min.   :   0.00
##  1st Qu.:   0.00
##  Median :   0.52
##  Mean   :  12.85
##  3rd Qu.:   6.57
##  Max.   :1090.84
```

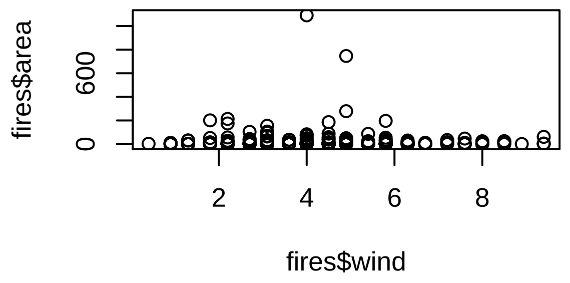
```
par(mfrow=c(2,2))
plot(fires$rain, fires$area)
plot(fires$temp, fires$area)
plot(fires$FFMC, fires$area)
plot(fires$DMC, fires$area)
```

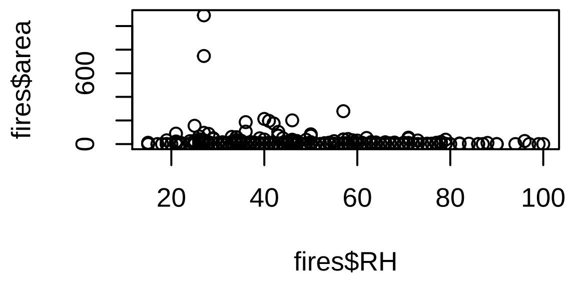


```
par(mfrow=c(2,2))
plot(fires$DC, fires$area)
plot(fires$ISI, fires$area)
plot(fires$wind, fires$area)
plot(fires$RH, fires$area)
```









Some summary statistics

```
summary(fires)
```

```
##        X               Y          month               day
##  Min.   :1.000   Min.   :2.0   Length:517         Length:517
##  1st Qu.:3.000   1st Qu.:4.0   Class :character   Class :character
##  Median :4.000   Median :4.0   Mode  :character   Mode  :character
##  Mean   :4.669   Mean   :4.3
##  3rd Qu.:7.000   3rd Qu.:5.0
##  Max.   :9.000   Max.   :9.0
##       FFMC            DMC              DC             ISI
##  Min.   :18.70   Min.   :  1.1   Min.   :  7.9   Min.   : 0.000
##  1st Qu.:90.20   1st Qu.: 68.6   1st Qu.:437.7   1st Qu.: 6.500
##  Median :91.60   Median :108.3   Median :664.2   Median : 8.400
##  Mean   :90.64   Mean   :110.9   Mean   :547.9   Mean   : 9.022
##  3rd Qu.:92.90   3rd Qu.:142.4   3rd Qu.:713.9   3rd Qu.:10.800
##  Max.   :96.20   Max.   :291.3   Max.   :860.6   Max.   :56.100
##       temp             RH              wind            rain
##  Min.   : 2.20   Min.   : 15.00   Min.   :0.400   Min.   :0.00000
##  1st Qu.:15.50   1st Qu.: 33.00   1st Qu.:2.700   1st Qu.:0.00000
##  Median :19.30   Median : 42.00   Median :4.000   Median :0.00000
##  Mean   :18.89   Mean   : 44.29   Mean   :4.018   Mean   :0.02166
##  3rd Qu.:22.80   3rd Qu.: 53.00   3rd Qu.:4.900   3rd Qu.:0.00000
##  Max.   :33.30   Max.   :100.00   Max.   :9.400   Max.   :6.40000
##       area
##  Min.   :   0.00
##  1st Qu.:   0.00
##  Median :   0.52
##  Mean   :  12.85
##  3rd Qu.:   6.57
##  Max.   :1090.84
```

```
describe(fires, skew = FALSE, omit = TRUE)
```

```
## Warning in describe(fires, skew = FALSE, omit = TRUE): NAs introduced by
## coercion

## Warning in describe(fires, skew = FALSE, omit = TRUE): NAs introduced by
## coercion

##      vars   n   mean     sd  min     max   range    se
## X       1 517   4.67   2.31  1.0    9.00    8.00  0.10
## Y       2 517   4.30   1.23  2.0    9.00    7.00  0.05
## FFMC    5 517  90.64   5.52 18.7   96.20   77.50  0.24
## DMC     6 517 110.87  64.05  1.1  291.30  290.20  2.82
## DC      7 517 547.94 248.07  7.9  860.60  852.70 10.91
## ISI     8 517   9.02   4.56  0.0   56.10   56.10  0.20
## temp    9 517  18.89   5.81  2.2   33.30   31.10  0.26
## RH     10 517  44.29  16.32 15.0  100.00   85.00  0.72
## wind   11 517   4.02   1.79  0.4    9.40    9.00  0.08
## rain   12 517   0.02   0.30  0.0    6.40    6.40  0.01
## area   13 517  12.85  63.66  0.0 1090.84 1090.84  2.80
```

```
table(fires$month)
```

```
##
## apr aug dec feb jan jul jun mar may nov oct sep
##   9 184   9  20   2  32  17  54   2   1  15 172
```

```
table(fires$day)
```

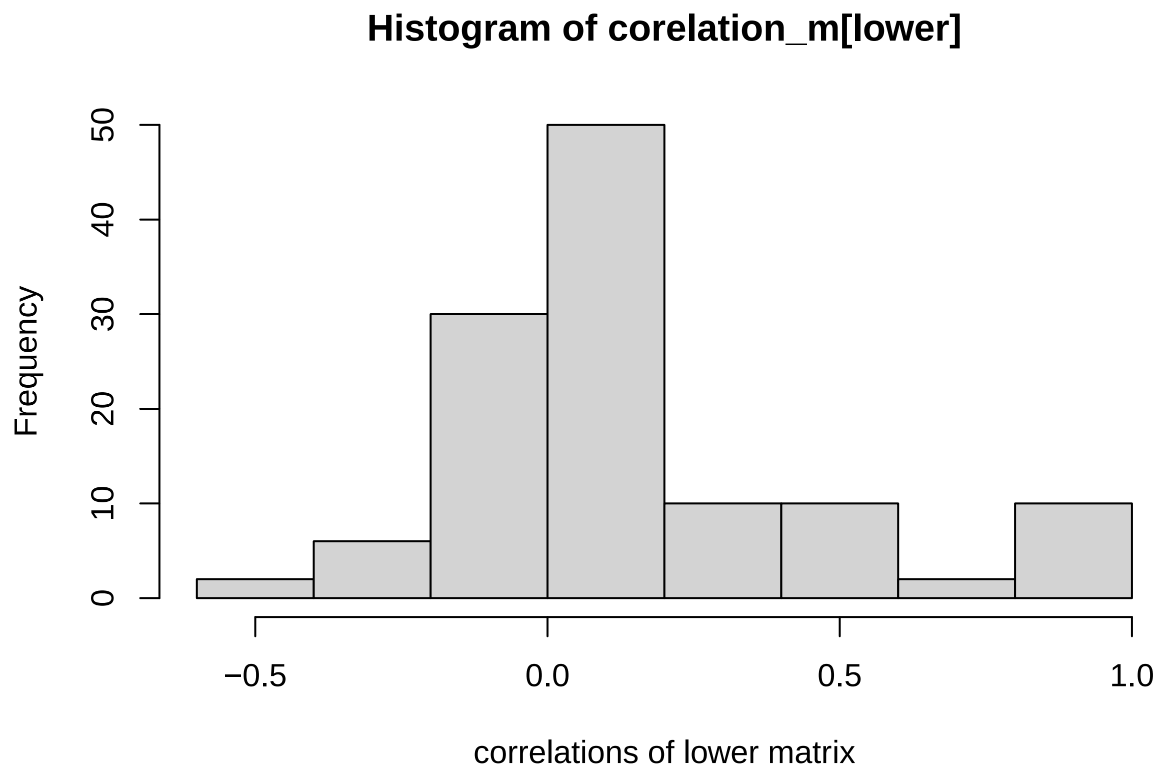
```
##
## fri mon sat sun thu tue wed
##  85  74  84  95  61  64  54
```

Correlation

```
fires_matirx = fires[,-c(3,4)]
corelation_m = cor(fires_matirx, use='pairwise.complete.obs')
lower = lower.tri(fires_matirx)
corelation_m
```

```
##                 X            Y        FFMC          DMC          DC
## X     1.000000000  0.539548171 -0.02103927 -0.048384178 -0.08591612
## Y     0.539548171  1.000000000 -0.04630755  0.007781561 -0.10117777
## FFMC -0.021039272 -0.046307546  1.00000000  0.382618800  0.33051180
## DMC  -0.048384178  0.007781561  0.38261880  1.000000000  0.68219161
## DC   -0.085916123 -0.101177767  0.33051180  0.682191612  1.00000000
## ISI   0.006209941 -0.024487992  0.53180493  0.305127835  0.22915417
## temp -0.051258262 -0.024103084  0.43153226  0.469593844  0.49620805
## RH    0.085223194  0.062220731 -0.30099542  0.073794941 -0.03919165
## wind  0.018797818 -0.020340852 -0.02848481 -0.105342253 -0.20346569
## rain  0.065387168  0.033234103  0.05670153  0.074789982  0.03586086
## area  0.063385299  0.044873225  0.04012200  0.072994296  0.04938323
##               ISI        temp          RH        wind         rain         area
## X     0.006209941 -0.05125826  0.08522319  0.01879782  0.065387168  0.063385299
## Y    -0.024487992 -0.02410308  0.06222073 -0.02034085  0.033234103  0.044873225
## FFMC  0.531804931  0.43153226 -0.30099542 -0.02848481  0.056701533  0.040122004
## DMC   0.305127835  0.46959384  0.07379494 -0.10534225  0.074789982  0.072994296
## DC    0.229154169  0.49620805 -0.03919165 -0.20346569  0.035860862  0.049383225
## ISI   1.000000000  0.39428710 -0.13251718  0.10682589  0.067668190  0.008257688
## temp  0.394287104  1.00000000 -0.52739034 -0.22711622  0.069490547  0.097844107
## RH   -0.132517177 -0.52739034  1.00000000  0.06941007  0.099751223 -0.075518563
## wind  0.106825888 -0.22711622  0.06941007  1.00000000  0.061118880  0.012317277
## rain  0.067668190  0.06949055  0.09975122  0.06111888  1.000000000 -0.007365729
## area  0.008257688  0.09784411 -0.07551856  0.01231728 -0.007365729  1.000000000
```

```
hist(corelation_m[lower], xlab = 'correlations of lower matrix')
```



This is for fires not using X,Y,Day,Month

```
fires_lm = log_fires[,-c(1,2,3,4)]
#head(log_fires_lm)
#head(fires_lm)
#fitting the most generic model
set.seed(1)
test = sample(nrow(fires_lm), floor(nrow(fires_lm)*.2))
linear_fit = lm(log_area~., data = fires_lm[-test,])
summary(linear_fit)
```

```
##
## Call:
## lm(formula = log_area ~ ., data = fires_lm[-test, ])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.5527 -1.1397 -0.5844  0.8969  5.6387
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.0541516  1.4660322  -0.037   0.9706
## FFMC         0.0080444  0.0154786   0.520   0.6035
## DMC          0.0009709  0.0017045   0.570   0.5692
## DC           0.0001425  0.0004062   0.351   0.7259
## ISI         -0.0256785  0.0183699  -1.398   0.1629
## temp         0.0175089  0.0198360   0.883   0.3779
## RH          -0.0022251  0.0059553  -0.374   0.7089
## wind         0.0736896  0.0422456   1.744   0.0819 .
## rain         0.0966483  0.2197509   0.440   0.6603
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.431 on 405 degrees of freedom
## Multiple R-squared:  0.02054,    Adjusted R-squared:  0.001196
## F-statistic: 1.062 on 8 and 405 DF,  p-value: 0.3891
```

```
# nothing significant

# getting MSE for the test
pred = predict(linear_fit, fires_lm[test,])
mean((pred - fires_lm[test, "log_area"])**2)
```

```
## [1] 1.578537
```

```
linear_fit = glm(log_area ~., data = fires_lm[-test,])
summary(linear_fit)
```

```
##
## Call:
## glm(formula = log_area ~ ., data = fires_lm[-test, ])
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5527  -1.1397  -0.5844   0.8969   5.6387
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.0541516  1.4660322  -0.037   0.9706
## FFMC         0.0080444  0.0154786   0.520   0.6035
## DMC          0.0009709  0.0017045   0.570   0.5692
## DC           0.0001425  0.0004062   0.351   0.7259
## ISI         -0.0256785  0.0183699  -1.398   0.1629
## temp         0.0175089  0.0198360   0.883   0.3779
## RH          -0.0022251  0.0059553  -0.374   0.7089
## wind         0.0736896  0.0422456   1.744   0.0819 .
## rain         0.0966483  0.2197509   0.440   0.6603
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 2.04787)
##
##     Null deviance: 846.78  on 413  degrees of freedom
## Residual deviance: 829.39  on 405  degrees of freedom
## AIC: 1482.5
##
## Number of Fisher Scoring iterations: 2
```

```
#10 CV validation for the training set
cv.glm(fires_lm[-test,], linear_fit, K = 10)$delta[1]
```

```
## [1] 2.105039
```

```
# getting MSE for the test
pred = predict(linear_fit, fires_lm[test,])
mean((pred - fires_lm[test, "log_area"])**2)
```

```
## [1] 1.578537
```

This is for fires not using Day,Month

```
fires_lm = log_fires[,-c(3,4)]
#head(fires_lm)
#fitting the most generic model
fires_lm = log_fires[,-c(3,4)]
#head(fires_lm)
#fitting the most generic model
linear_fit = lm(log_area~., data = fires_lm[-test,])
linear_fit = glm(log_area ~ ., data = fires_lm[-test,])
summary(linear_fit) # wind significant
```

```
##
## Call:
## glm(formula = log_area ~ ., data = fires_lm[-test, ])
##
## Deviance Residuals:
```

```
##     Min       1Q   Median       3Q      Max
## -1.6812  -1.1236  -0.5763   0.8849   5.5580
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.2660400  1.4945471  -0.178    0.859
## X            0.0436578  0.0365873   1.193    0.233
## Y            0.0188301  0.0687325   0.274    0.784
## FFMC         0.0074528  0.0154887   0.481    0.631
## DMC          0.0010036  0.0017158   0.585    0.559
## DC           0.0002053  0.0004104   0.500    0.617
## ISI         -0.0259818  0.0183641  -1.415    0.158
## temp         0.0159706  0.0198453   0.805    0.421
## RH          -0.0029745  0.0059704  -0.498    0.619
## wind         0.0752684  0.0423099   1.779    0.076 .
## rain         0.0788902  0.2199044   0.359    0.720
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 2.04499)
##
##     Null deviance: 846.78  on 413  degrees of freedom
## Residual deviance: 824.13  on 403  degrees of freedom
## AIC: 1483.9
##
## Number of Fisher Scoring iterations: 2
```

```
#10 CV validation for the training set
cv.glm(fires_lm[-test,], linear_fit, K = 10)$delta[1]
```

```
## [1] 2.188298
```

```
# getting MSE for the test
pred = predict(linear_fit, fires_lm[test,])
mean((pred - fires_lm[test, "log_area"])**2)
```

```
## [1] 1.57482
```

```
log_fires$month = as.factor(log_fires$month)
log_fires$day = as.factor(log_fires$day)

newdata = one_hot(as.data.table(log_fires))
newdata
```

```
##      X Y month_apr month_aug month_dec month_feb month_jan month_jul month_jun
##   1: 7 5         0         0         0         0         0         0         0
##   2: 7 4         0         0         0         0         0         0         0
##   3: 7 4         0         0         0         0         0         0         0
##   4: 8 6         0         0         0         0         0         0         0
##   5: 8 6         0         0         0         0         0         0         0
##  ---
## 513: 4 3         0         1         0         0         0         0         0
## 514: 2 4         0         1         0         0         0         0         0
## 515: 7 4         0         1         0         0         0         0         0
## 516: 1 4         0         1         0         0         0         0         0
## 517: 6 3         0         0         0         0         0         0         0
##      month_mar month_may month_nov month_oct month_sep day_fri day_mon day_sat
##   1:         1         0         0         0         0       1       0       0
##   2:         0         0         0         1         0       0       0       0
##   3:         0         0         0         1         0       0       0       1
##   4:         1         0         0         0         0       1       0       0
##   5:         1         0         0         0         0       0       0       0
##  ---
## 513:         0         0         0         0         0       0       0       0
## 514:         0         0         0         0         0       0       0       0
## 515:         0         0         0         0         0       0       0       0
## 516:         0         0         0         0         0       0       0       1
## 517:         0         0         1         0         0       0       0       0
##      day_sun day_thu day_tue day_wed FFMC   DMC    DC  ISI temp RH wind rain
##   1:       0       0       0       0 86.2  26.2  94.3  5.1  8.2 51  6.7  0.0
##   2:       0       0       1       0 90.6  35.4 669.1  6.7 18.0 33  0.9  0.0
##   3:       0       0       0       0 90.6  43.7 686.9  6.7 14.6 33  1.3  0.0
##   4:       0       0       0       0 91.7  33.3  77.5  9.0  8.3 97  4.0  0.2
##   5:       1       0       0       0 89.3  51.3 102.2  9.6 11.4 99  1.8  0.0
##  ---
## 513:       1       0       0       0 81.6  56.7 665.6  1.9 27.8 32  2.7  0.0
## 514:       1       0       0       0 81.6  56.7 665.6  1.9 21.9 71  5.8  0.0
## 515:       1       0       0       0 81.6  56.7 665.6  1.9 21.2 70  6.7  0.0
## 516:       0       0       0       0 94.4 146.0 614.7 11.3 25.6 42  4.0  0.0
## 517:       0       0       1       0 79.5   3.0 106.7  1.1 11.8 31  4.5  0.0
##      log_area
##   1: 0.000000
##   2: 0.000000
##   3: 0.000000
##   4: 0.000000
##   5: 0.000000
##  ---
## 513: 2.006871
## 514: 4.012592
## 515: 2.498152
## 516: 0.000000
## 517: 0.000000
```

```
linear_fit = glm(log_area~., data = newdata[-test,])
summary(linear_fit) # month_dec significant, but only 9 instances
```

```
##
## Call:
## glm(formula = log_area ~ ., data = newdata[-test, ])
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0198  -1.0467  -0.4367   0.8368   5.2292
##
## Coefficients: (2 not defined because of singularities)
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.497993   2.295168  -0.217  0.82834
## X            0.049443   0.037327   1.325  0.18609
## Y           -0.018272   0.070122  -0.261  0.79455
## month_apr   -0.336320   1.046950  -0.321  0.74820
## month_aug   -0.736235   0.258725  -2.846  0.00467 **
## month_dec    1.327058   0.834355   1.591  0.11254
## month_feb   -0.600055   0.939059  -0.639  0.52320
## month_jan   -0.012159   2.121344  -0.006  0.99543
## month_jul   -0.782580   0.494872  -1.581  0.11461
## month_jun   -0.899796   0.726886  -1.238  0.21651
## month_mar   -1.064230   0.880766  -1.208  0.22767
## month_may   -0.110310   1.307457  -0.084  0.93281
## month_nov   -1.831009   1.666084  -1.099  0.27246
## month_oct    0.170648   0.476787   0.358  0.72060
## month_sep          NA         NA      NA       NA
## day_fri     -0.397563   0.285532  -1.392  0.16462
## day_mon     -0.134210   0.295006  -0.455  0.64941
## day_sat      0.041814   0.285995   0.146  0.88384
## day_sun     -0.167346   0.277776  -0.602  0.54723
## day_thu     -0.421367   0.299179  -1.408  0.15981
## day_tue      0.045127   0.300072   0.150  0.88054
## day_wed            NA         NA      NA       NA
## FFMC         0.015902   0.021408   0.743  0.45805
## DMC          0.004577   0.002168   2.111  0.03543 *
## DC          -0.001740   0.001415  -1.229  0.21969
## ISI         -0.019658   0.020283  -0.969  0.33305
## temp         0.046407   0.025356   1.830  0.06799 .
## RH           0.002568   0.007027   0.365  0.71501
## wind         0.076482   0.044422   1.722  0.08592 .
## rain         0.040980   0.222596   0.184  0.85403
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 2.011559)
##
##     Null deviance: 846.78  on 413  degrees of freedom
## Residual deviance: 776.46  on 386  degrees of freedom
## AIC: 1493.2
##
## Number of Fisher Scoring iterations: 2
```

```
#10 CV validation for the training set
cv.glm(fires_lm[-test,], linear_fit, K = 10)$delta[1]
```

```
## [1] 2.127222
```

```
# getting MSE for the test
pred = predict(linear_fit, newdata[test,])
```

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

```r
mean((pred - fires_lm[test, "log_area"])**2)
```

```
## [1] 1.649722
```

Trying different sub-selection models now

```r
# Set seed for reproducibility
set.seed(123)
# Set up repeated k-fold cross-validation
train.control <- trainControl(method = "cv", number = 10)
# Train the model
step.model <- train(log_area ~., data = newdata[-test,],
                    method = "leapForward",
                    tuneGrid = data.frame(nvmax = 1:28),
                    trControl = train.control
                    )
```

```
## Warning in leaps.setup(x, y, wt = weights, nbest = nbest, nvmax = nvmax, : 2
## linear dependencies found
```

```
## Reordering variables and trying again:
```

```
## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to replace is
## not a multiple of replacement length
```

```
## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: 2 linear dependencies found
```

```
## Reordering variables and trying again:
```

```
## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to replace is
## not a multiple of replacement length
```

```
## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: 2 linear dependencies found
```

```
## Reordering variables and trying again:
```

```
## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to replace is
## not a multiple of replacement length
```

```
## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: 2 linear dependencies found
```

```
## Reordering variables and trying again:
```

```
## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to replace is
## not a multiple of replacement length
```

```
## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: 2 linear dependencies found
```

```
## Reordering variables and trying again:
```

```
## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to replace is
## not a multiple of replacement length

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: 2 linear dependencies found

## Reordering variables and trying again:

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to replace is
## not a multiple of replacement length

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: 2 linear dependencies found

## Reordering variables and trying again:

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to replace is
## not a multiple of replacement length

## Warning in leaps.setup(x, y, wt = weights, nbest = nbest, nvmax = nvmax, : 4
## linear dependencies found

## Reordering variables and trying again:

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to replace is
## not a multiple of replacement length

## Warning: predictions failed for Fold08: nvmax=28 Error in method$predict(modelFit = modelFit, newdat
##   Some values of 'nvmax' are not in the model sequence.

## Warning in leaps.setup(x, y, wt = weights, nbest = nbest, nvmax = nvmax, : 2
## linear dependencies found

## Reordering variables and trying again:

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to replace is
## not a multiple of replacement length

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: 2 linear dependencies found

## Reordering variables and trying again:

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to replace is
## not a multiple of replacement length

## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.

## Warning in leaps.setup(x, y, wt = weights, nbest = nbest, nvmax = nvmax, : 2
## linear dependencies found

## Reordering variables and trying again:
```

```
step.model$results
```

```
##    nvmax     RMSE   Rsquared      MAE     RMSESD RsquaredSD      MAESD
## 1      1 1.442623 0.01906133 1.179176 0.11129566 0.01598865 0.05480331
## 2      2 1.437062 0.01551776 1.169685 0.11422318 0.01167727 0.05631316
## 3      3 1.450492 0.01802989 1.177800 0.10610337 0.02083837 0.05453590
## 4      4 1.451807 0.02123694 1.177778 0.11303794 0.02916368 0.06064563
## 5      5 1.457856 0.01706735 1.185396 0.10035258 0.01726989 0.05891346
## 6      6 1.463036 0.01669974 1.189095 0.10225319 0.02122934 0.06350963
## 7      7 1.462220 0.01235698 1.187877 0.10451994 0.01684824 0.06523244
## 8      8 1.463280 0.01595948 1.188638 0.10116481 0.02082390 0.06659032
## 9      9 1.459806 0.01868669 1.183833 0.10263248 0.02193200 0.06811831
## 10    10 1.462937 0.01903391 1.185934 0.10660041 0.02122975 0.07172406
## 11    11 1.461124 0.02019685 1.178437 0.10524120 0.02287606 0.06712397
## 12    12 1.458853 0.02209197 1.177172 0.10209893 0.02394473 0.06443968
## 13    13 1.463784 0.02934464 1.177641 0.09791089 0.03695111 0.06990224
## 14    14 1.464009 0.02829471 1.176853 0.09163130 0.03659793 0.06315348
## 15    15 1.460381 0.02967333 1.175263 0.09084583 0.03693763 0.06029802
## 16    16 1.477373 0.02762074 1.183388 0.10521202 0.02624631 0.06885729
## 17    17 1.489269 0.01959249 1.188606 0.08500601 0.02339080 0.05554852
## 18    18 1.489397 0.01750349 1.189290 0.09372076 0.01831320 0.05969218
## 19    19 1.486032 0.01696212 1.192690 0.08010966 0.01913090 0.05685590
## 20    20 1.501405 0.01748066 1.202640 0.10653034 0.01775052 0.06888272
## 21    21 1.501535 0.02011308 1.201299 0.11068866 0.01786742 0.07496236
## 22    22 1.497299 0.01955870 1.201933 0.10912724 0.01774629 0.07825362
## 23    23 1.498593 0.01954569 1.202251 0.10756352 0.01795036 0.07904081
## 24    24 1.502652 0.01953948 1.203645 0.10492533 0.01812450 0.07759338
## 25    25 1.500869 0.02241965 1.204158 0.11555147 0.02173852 0.08565613
## 26    26 1.507730 0.02088676 1.210301 0.11952470 0.02283717 0.08347328
## 27    27 1.511937 0.02176172 1.215792 0.11740105 0.02042992 0.08353945
## 28    28 1.511937 0.02176172 1.215792 0.11740105 0.02042992 0.08353945
```

```
step.model$bestTune
```

```
##   nvmax
## 2     2
```

```
summary(step.model$finalModel)
```

```
## Subset selection object
## 29 Variables  (and intercept)
##           Forced in Forced out
## X             FALSE      FALSE
## Y             FALSE      FALSE
## month_apr     FALSE      FALSE
## month_aug     FALSE      FALSE
## month_dec     FALSE      FALSE
## month_feb     FALSE      FALSE
## month_jan     FALSE      FALSE
## month_jul     FALSE      FALSE
## month_jun     FALSE      FALSE
## month_mar     FALSE      FALSE
## month_may     FALSE      FALSE
## month_nov     FALSE      FALSE
## month_oct     FALSE      FALSE
## day_fri       FALSE      FALSE
## day_mon       FALSE      FALSE
## day_sat       FALSE      FALSE
## day_sun       FALSE      FALSE
## day_thu       FALSE      FALSE
## day_tue       FALSE      FALSE
## FFMC          FALSE      FALSE
## DMC           FALSE      FALSE
## DC            FALSE      FALSE
## ISI           FALSE      FALSE
## temp          FALSE      FALSE
## RH            FALSE      FALSE
## wind          FALSE      FALSE
## rain          FALSE      FALSE
## month_sep     FALSE      FALSE
## day_wed       FALSE      FALSE
## 1 subsets of each size up to 3
## Selection Algorithm: forward
##          X   Y   month_apr month_aug month_dec month_feb month_jan month_jul
## 1  ( 1 ) " " " " " "       " "       "*"       " "       " "       " "      
## 2  ( 1 ) " " " " " "       " "       "*"       " "       " "       " "      
## 3  ( 1 ) " " " " " "       "*"       "*"       " "       " "       " "      
##          month_jun month_mar month_may month_nov month_oct month_sep day_fri
## 1  ( 1 ) " "       " "       " "       " "       " "       " "       " "    
## 2  ( 1 ) " "       " "       " "       " "       " "       " "       " "    
## 3  ( 1 ) " "       " "       " "       " "       " "       " "       " "    
##          day_mon day_sat day_sun day_thu day_tue day_wed FFMC DMC DC  ISI temp
## 1  ( 1 ) " "     " "     " "     " "     " "     " "     " "  " " " " " " " " 
## 2  ( 1 ) " "     " "     " "     " "     " "     " "     " "  " " " " " " "*" 
## 3  ( 1 ) " "     " "     " "     " "     " "     " "     " "  " " " " " " "*" 
##          RH  wind rain
## 1  ( 1 ) " " " "  " " 
## 2  ( 1 ) " " " "  " " 
## 3  ( 1 ) " " " "  " " 
```

Using the above forward selection method we got that temp was the best forward selection model. Here is code of us actually checking the MSE for temp

```r
linear_fit = glm(log_area~month_dec, data = newdata[-test,])

# getting MSE for the test
pred = predict(linear_fit, newdata[test,])
mean((pred - fires_lm[test, "log_area"])**2)
```

```
## [1] 1.49972
```

```r
# Set seed for reproducibility
set.seed(123)
# Set up repeated k-fold cross-validation
train.control <- trainControl(method = "cv", number = 10)
```

```r
# Train the model
step.model <- train(log_area ~., data = newdata[-test,],
                    method = "leapBackward",
                    tuneGrid = data.frame(nvmax = 1:28),
                    trControl = train.control
                    )
```

```
## Warning in leaps.setup(x, y, wt = weights, nbest = nbest, nvmax = nvmax, : 2
## linear dependencies found

## Reordering variables and trying again:

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to replace is
## not a multiple of replacement length

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: 2 linear dependencies found

## Reordering variables and trying again:

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to replace is
## not a multiple of replacement length

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: 2 linear dependencies found

## Reordering variables and trying again:

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to replace is
## not a multiple of replacement length

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: 2 linear dependencies found

## Reordering variables and trying again:

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to replace is
## not a multiple of replacement length

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: 2 linear dependencies found

## Reordering variables and trying again:

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to replace is
## not a multiple of replacement length

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: 2 linear dependencies found

## Reordering variables and trying again:

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to replace is
## not a multiple of replacement length

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: 2 linear dependencies found
## Reordering variables and trying again:

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to replace is
## not a multiple of replacement length

## Warning in leaps.setup(x, y, wt = weights, nbest = nbest, nvmax = nvmax, : 4
## linear dependencies found

## Reordering variables and trying again:

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to replace is
## not a multiple of replacement length

## Warning: predictions failed for Fold08: nvmax=28 Error in method$predict(modelFit = modelFit, newdat
##   Some values of 'nvmax' are not in the model sequence.

## Warning in leaps.setup(x, y, wt = weights, nbest = nbest, nvmax = nvmax, : 2
## linear dependencies found

## Reordering variables and trying again:

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to replace is
## not a multiple of replacement length

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: 2 linear dependencies found

## Reordering variables and trying again:

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to replace is
## not a multiple of replacement length

## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.

## Warning in leaps.setup(x, y, wt = weights, nbest = nbest, nvmax = nvmax, : 2
## linear dependencies found

## Reordering variables and trying again:
```

```
step.model$results
```

```
##    nvmax     RMSE    Rsquared      MAE     RMSESD  RsquaredSD      MAESD
## 1      1 1.441481 0.009349211 1.179260 0.10641031 0.008620743 0.05065523
## 2      2 1.433722 0.016020733 1.168274 0.10885170 0.011037346 0.05382705
## 3      3 1.433786 0.018971258 1.167176 0.10785957 0.020157989 0.05437317
## 4      4 1.439834 0.022238721 1.173448 0.12467065 0.030429024 0.06289880
## 5      5 1.442508 0.023310163 1.177713 0.11886355 0.030152643 0.06899525
## 6      6 1.446295 0.022909969 1.182996 0.11972505 0.028701879 0.07059269
## 7      7 1.457435 0.013930270 1.191232 0.10998885 0.020382654 0.05955340
## 8      8 1.453353 0.023018401 1.182990 0.09923473 0.026823384 0.05774209
## 9      9 1.450926 0.026493453 1.181954 0.10129919 0.026891858 0.05897348
## 10    10 1.451456 0.025717609 1.184278 0.09921976 0.025869497 0.06195397
## 11    11 1.450324 0.025051225 1.183258 0.10129265 0.025769714 0.06299833
## 12    12 1.445272 0.017655210 1.174997 0.10293532 0.019214183 0.06478342
## 13    13 1.458958 0.013791105 1.183381 0.10294732 0.016757804 0.06750569
## 14    14 1.468206 0.011833509 1.191410 0.11068161 0.014778650 0.07269582
## 15    15 1.477275 0.009162093 1.200177 0.09573290 0.013204178 0.06898456
## 16    16 1.475965 0.010529817 1.197912 0.09078564 0.013998826 0.06629562
## 17    17 1.469963 0.011127206 1.194391 0.09498333 0.014329633 0.06107032
## 18    18 1.504709 0.012585632 1.210008 0.11508980 0.012654011 0.07983149
## 19    19 1.504279 0.012622825 1.208995 0.11346371 0.012466354 0.07740750
## 20    20 1.503907 0.012276394 1.210440 0.11479531 0.014454677 0.07726371
## 21    21 1.505830 0.012592000 1.213441 0.10886321 0.015022391 0.07874978
## 22    22 1.518146 0.010977859 1.219073 0.12614765 0.012077133 0.08290400
## 23    23 1.520311 0.012299175 1.219170 0.12588763 0.012782838 0.08439579
## 24    24 1.519827 0.012106078 1.219429 0.12375168 0.013528588 0.08279455
## 25    25 1.521031 0.012539499 1.221069 0.12531692 0.014106047 0.08553048
## 26    26 1.520983 0.013959477 1.221487 0.12613546 0.014250407 0.08572706
## 27    27 1.527126 0.018147063 1.225386 0.12815885 0.019568806 0.08991750
## 28    28 1.527126 0.018147063 1.225386 0.12815885 0.019568806 0.08991750
```

```r
step.model$bestTune
```

```
##   nvmax
## 2     2
```

```r
summary(step.model$finalModel)
```

```
## Subset selection object
## 29 Variables  (and intercept)
##           Forced in Forced out
## X             FALSE      FALSE
## Y             FALSE      FALSE
## month_apr     FALSE      FALSE
## month_aug     FALSE      FALSE
## month_dec     FALSE      FALSE
## month_feb     FALSE      FALSE
## month_jan     FALSE      FALSE
## month_jul     FALSE      FALSE
## month_jun     FALSE      FALSE
## month_mar     FALSE      FALSE
## month_may     FALSE      FALSE
## month_nov     FALSE      FALSE
## month_oct     FALSE      FALSE
## day_fri       FALSE      FALSE
## day_mon       FALSE      FALSE
## day_sat       FALSE      FALSE
## day_sun       FALSE      FALSE
## day_thu       FALSE      FALSE
## day_tue       FALSE      FALSE
## FFMC          FALSE      FALSE
## DMC           FALSE      FALSE
## DC            FALSE      FALSE
## ISI           FALSE      FALSE
## temp          FALSE      FALSE
## RH            FALSE      FALSE
## wind          FALSE      FALSE
## rain          FALSE      FALSE
## month_sep     FALSE      FALSE
## day_wed       FALSE      FALSE
## 1 subsets of each size up to 3
## Selection Algorithm: backward
##          X   Y   month_apr month_aug month_dec month_feb month_jan month_jul
## 1  ( 1 ) " " " " " "       " "       "*"       " "       " "       " "
## 2  ( 1 ) " " " " " "       " "       "*"       " "       " "       " "
## 3  ( 1 ) " " " " " "       "*"       "*"       " "       " "       " "
##          month_jun month_mar month_may month_nov month_oct month_sep day_fri
## 1  ( 1 ) " "       " "       " "       " "       " "       " "       " "
## 2  ( 1 ) " "       " "       " "       " "       " "       " "       " "
## 3  ( 1 ) " "       " "       " "       " "       " "       " "       " "
##          day_mon day_sat day_sun day_thu day_tue day_wed FFMC DMC DC  ISI temp
## 1  ( 1 ) " "     " "     " "     " "     " "     " "     " "  " " " " " " " "
## 2  ( 1 ) " "     " "     " "     " "     " "     " "     " "  " " " " " " "*"
## 3  ( 1 ) " "     " "     " "     " "     " "     " "     " "  " " " " " " "*"
##          RH  wind rain
## 1  ( 1 ) " " " "  " "
## 2  ( 1 ) " " " "  " "
## 3  ( 1 ) " " " "  " "
```

Backwards selection choose the model with 1 variables - month dec

```
linear_fit = glm(log_area~month_dec, data = newdata[-test,])
summary(linear_fit)
```

```
##
## Call:
## glm(formula = log_area ~ month_dec, data = newdata[-test, ])
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.1373  -1.1373  -0.5984   0.8796   5.8583
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.13733    0.07055  16.121   <2e-16 ***
## month_dec    1.31183    0.58601   2.239   0.0257 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 2.0306)
##
##     Null deviance: 846.78  on 413  degrees of freedom
## Residual deviance: 836.61  on 412  degrees of freedom
## AIC: 1472.1
##
## Number of Fisher Scoring iterations: 2
```

```
# getting MSE for the test
pred = predict(linear_fit, newdata[test,])
mean((pred - fires_lm[test, "log_area"])**2)
```

```
## [1] 1.49972
```

```
# Set seed for reproducibility
set.seed(123)
# Set up repeated k-fold cross-validation
train.control <- trainControl(method = "cv", number = 10)
# Train the model
step.model <- train(log_area ~., data = newdata[-test,],
                    method = "leapSeq",
                    tuneGrid = data.frame(nvmax = 1:20),
                    trControl = train.control
                    )
```

```
## Warning in leaps.setup(x, y, wt = weights, nbest = nbest, nvmax = nvmax, : 2
## linear dependencies found

## Reordering variables and trying again:

## Warning in leaps.setup(x, y, wt = weights, nbest = nbest, nvmax = nvmax, : 2
## linear dependencies found

## Reordering variables and trying again:

## Warning in leaps.setup(x, y, wt = weights, nbest = nbest, nvmax = nvmax, : 2
## linear dependencies found

## Reordering variables and trying again:

## Warning in leaps.setup(x, y, wt = weights, nbest = nbest, nvmax = nvmax, : 2
## linear dependencies found

## Reordering variables and trying again:

## Warning in leaps.setup(x, y, wt = weights, nbest = nbest, nvmax = nvmax, : 2
## linear dependencies found

## Reordering variables and trying again:

## Warning in leaps.setup(x, y, wt = weights, nbest = nbest, nvmax = nvmax, : 2
## linear dependencies found

## Reordering variables and trying again:

## Warning in leaps.setup(x, y, wt = weights, nbest = nbest, nvmax = nvmax, : 2
## linear dependencies found
## Reordering variables and trying again:

## Warning in leaps.setup(x, y, wt = weights, nbest = nbest, nvmax = nvmax, : 4
## linear dependencies found

## Reordering variables and trying again:

## Warning in leaps.setup(x, y, wt = weights, nbest = nbest, nvmax = nvmax, : 2
## linear dependencies found

## Reordering variables and trying again:

## Warning in leaps.setup(x, y, wt = weights, nbest = nbest, nvmax = nvmax, : 2
## linear dependencies found

## Reordering variables and trying again:

## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.

## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## 2 linear dependencies found

## Reordering variables and trying again:
```

```
step.model$results
```

```
##    nvmax     RMSE   Rsquared      MAE     RMSESD RsquaredSD      MAESD
## 1      1 1.427032 0.01906133 1.175923 0.11593717 0.01598865 0.05268337
## 2      2 1.418975 0.01614697 1.164365 0.11271032 0.01091076 0.05208121
## 3      3 1.435912 0.02750912 1.179667 0.10033900 0.03302521 0.05163921
## 4      4 1.443273 0.02410846 1.185590 0.11688243 0.02784609 0.06538489
## 5      5 1.439134 0.03278439 1.183100 0.09236943 0.03390760 0.06363526
## 6      6 1.457760 0.03020250 1.197359 0.09674325 0.03742317 0.06597372
## 7      7 1.451011 0.01974945 1.190802 0.10920497 0.03476967 0.07454142
## 8      8 1.459605 0.02396419 1.195470 0.09785285 0.03177975 0.06933855
## 9      9 1.455898 0.03077699 1.193087 0.09766677 0.04014824 0.06912253
## 10    10 1.457788 0.02343544 1.192943 0.09895096 0.02332712 0.06956685
## 11    11 1.459494 0.02408903 1.189385 0.10173045 0.02563652 0.07147728
## 12    12 1.453241 0.02661612 1.186718 0.09665071 0.02564452 0.06713443
## 13    13 1.464675 0.02804031 1.191555 0.10126077 0.03787899 0.07559927
## 14    14 1.459876 0.03491157 1.187086 0.08371761 0.03882801 0.06721598
## 15    15 1.456695 0.03086958 1.186715 0.08872065 0.03252774 0.06690327
## 16    16 1.476973 0.03395991 1.194522 0.10792439 0.02859721 0.07662503
## 17    17 1.471166 0.02841828 1.197454 0.07014452 0.02362204 0.06022952
## 18    18 1.480855 0.01917052 1.193330 0.10558675 0.01936754 0.06426832
## 19    19 1.485312 0.02190160 1.196798 0.11643020 0.01867886 0.07695762
## 20    20 1.491738 0.01876158 1.202936 0.11514088 0.01936339 0.07809056
```

```r
step.model$bestTune
```

```
##   nvmax
## 2     2
```

```r
summary(step.model$finalModel)
```

```
## Subset selection object
## 29 Variables  (and intercept)
##           Forced in Forced out
## X             FALSE      FALSE
## Y             FALSE      FALSE
## month_apr     FALSE      FALSE
## month_aug     FALSE      FALSE
## month_dec     FALSE      FALSE
## month_feb     FALSE      FALSE
## month_jan     FALSE      FALSE
## month_jul     FALSE      FALSE
## month_jun     FALSE      FALSE
## month_mar     FALSE      FALSE
## month_may     FALSE      FALSE
## month_nov     FALSE      FALSE
## month_oct     FALSE      FALSE
## day_fri       FALSE      FALSE
## day_mon       FALSE      FALSE
## day_sat       FALSE      FALSE
## day_sun       FALSE      FALSE
## day_thu       FALSE      FALSE
## day_tue       FALSE      FALSE
## FFMC          FALSE      FALSE
## DMC           FALSE      FALSE
## DC            FALSE      FALSE
## ISI           FALSE      FALSE
## temp          FALSE      FALSE
## RH            FALSE      FALSE
## wind          FALSE      FALSE
## rain          FALSE      FALSE
## month_sep     FALSE      FALSE
## day_wed       FALSE      FALSE
## 1 subsets of each size up to 3
## Selection Algorithm: 'sequential replacement'
##          X   Y   month_apr month_aug month_dec month_feb month_jan month_jul
## 1  ( 1 ) " " " " " "       " "       "*"       " "       " "       " "
## 2  ( 1 ) " " " " " "       " "       "*"       " "       " "       " "
## 3  ( 1 ) " " " " " "       "*"       "*"       " "       " "       " "
##          month_jun month_mar month_may month_nov month_oct month_sep day_fri
## 1  ( 1 ) " "       " "       " "       " "       " "       " "       " "
## 2  ( 1 ) " "       " "       " "       " "       " "       " "       " "
## 3  ( 1 ) " "       " "       " "       " "       " "       " "       " "
##          day_mon day_sat day_sun day_thu day_tue day_wed FFMC DMC DC  ISI temp
## 1  ( 1 ) " "     " "     " "     " "     " "     " "     " "  " " " " " " " "
## 2  ( 1 ) " "     " "     " "     " "     " "     " "     " "  " " " " " " "*"
## 3  ( 1 ) " "      " "      " "      " "      " "      " "      " "  " " " " " " "*"
##          RH  wind rain
## 1  ( 1 ) " " " "  " "
## 2  ( 1 ) " " " "  " "
## 3  ( 1 ) " " " "  " "
```

We get out the 2 variable model with month_dec and temp, just like forward selection.

I would like to explore some interaction terms and maybe investigating more into the correlated variables

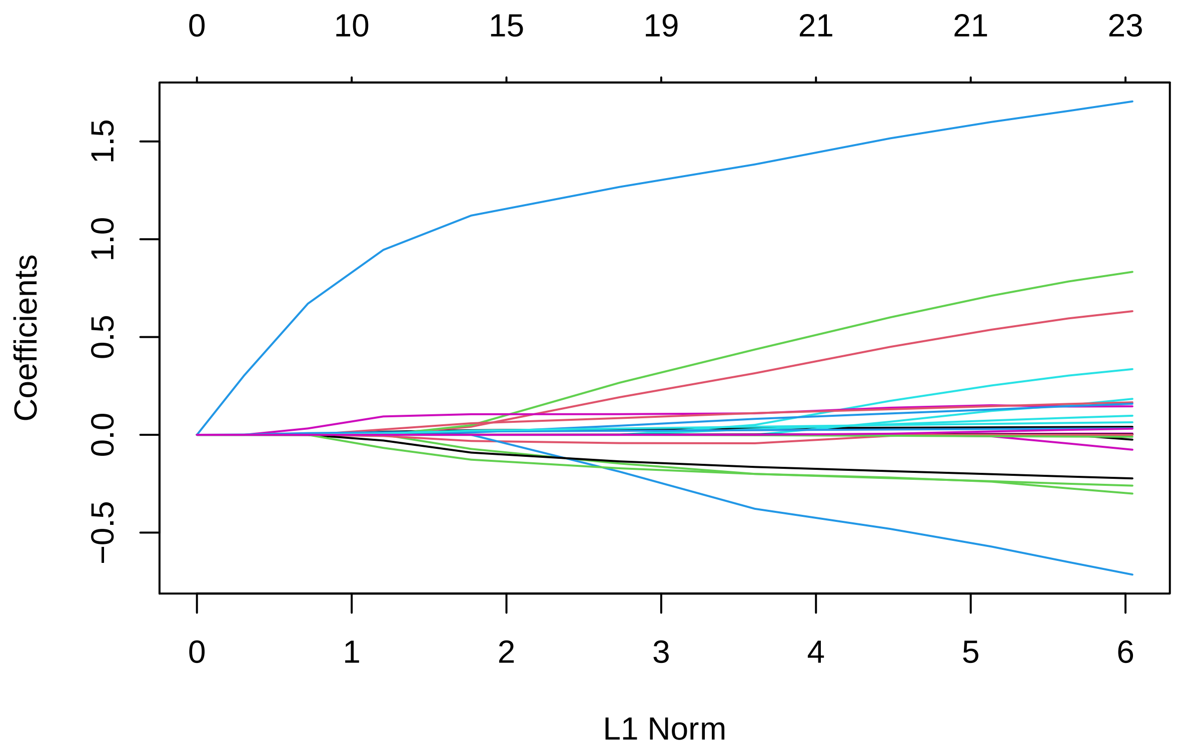
```
summary(lm(log_area ~ temp*day_sat, newdata))
```

```
##
## Call:
## lm(formula = log_area ~ temp * day_sat, data = newdata)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.3464 -1.0951 -0.7232  0.9227  5.7056
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.847332   0.225282   3.761 0.000189 ***
## temp          0.012836   0.011436   1.122 0.262238
## day_sat       0.165015   0.616345   0.268 0.789013
## temp:day_sat -0.001773   0.030784  -0.058 0.954090
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.4 on 513 degrees of freedom
## Multiple R-squared:  0.004059,   Adjusted R-squared:  -0.001765
## F-statistic: 0.697 on 3 and 513 DF,  p-value: 0.5542
```

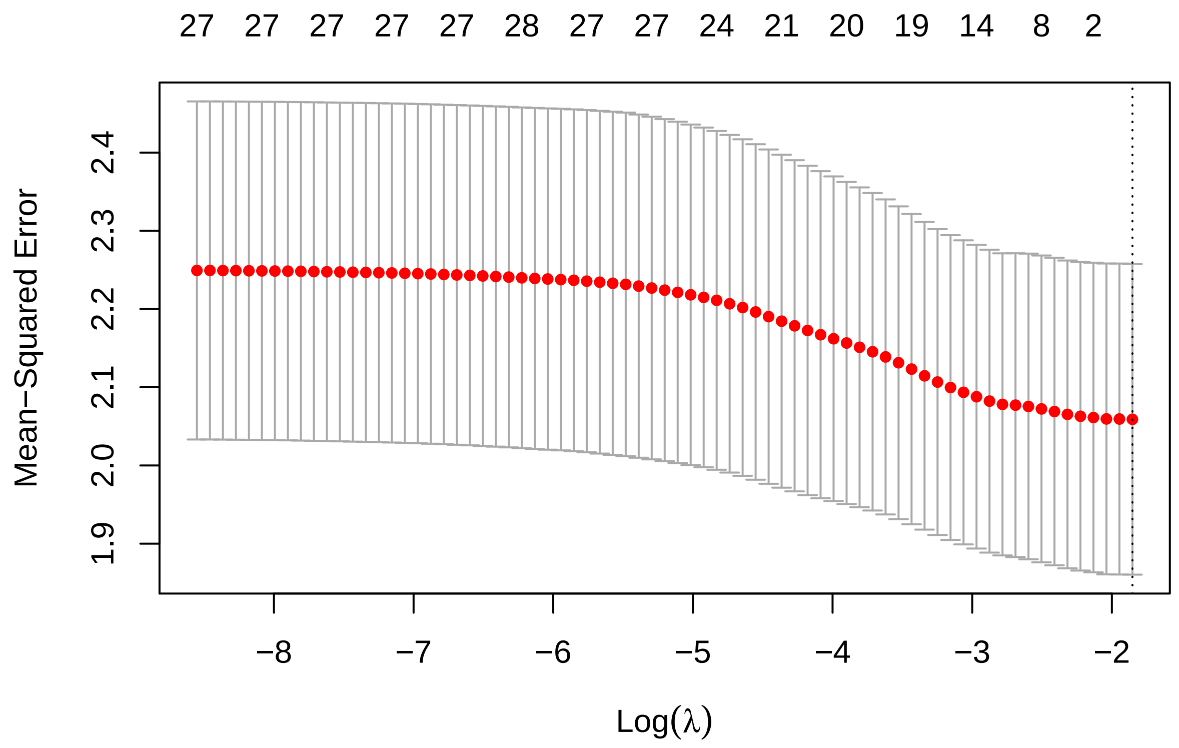
Lasso

```
grid=10^seq(10,-2,length=100)
lasso_mod = glmnet(x = as.matrix(newdata[-test, -"log_area"]), y = as.matrix(newdata[-test, log_area]),
plot(lasso_mod)
```

```
## Warning in regularize.values(x, y, ties, missing(ties), na.rm = na.rm):
## collapsing to unique 'x' values
```



```r
# cross validation
cv_lasso = cv.glmnet(x = as.matrix(newdata[-test, -"log_area"]), y = as.matrix(newdata[-test, log_area]
plot(cv_lasso)
```



```
bestlam = cv_lasso$lambda.min
lasso_pred=predict(lasso_mod,s=bestlam, newx=as.matrix(newdata[-test, -"log_area"]))
mean((lasso_pred-as.matrix(newdata[-test, log_area]))^2 - as.matrix(newdata[-test, log_area]))
```

```
## [1] 0.8869355
```

```
out=glmnet(as.matrix(newdata[, -"log_area"]), as.matrix(newdata[, log_area]), alpha=1,lambda=grid)
lasso_coef=predict(out,type="coefficients",s=bestlam)
lasso_coef
```

```
## 30 x 1 sparse Matrix of class "dgCMatrix"
##                     1
## (Intercept) 1.1060156
## X           .
## Y           .
## month_apr   .
## month_aug   .
## month_dec   0.2878068
## month_feb   .
## month_jan   .
## month_jul   .
## month_jun   .
## month_mar   .
## month_may   .
## month_nov   .
## month_oct   .
## month_sep   .
## day_fri     .
## day_mon     .
## day_sat     .
## day_sun     .
## day_thu     .
## day_tue     .
## day_wed     .
## FFMC        .
## DMC         .
## DC          .
## ISI         .
## temp        .
## RH          .
## wind        .
## rain        .
```

try different coding method

As described in hw2 we can combine days to weekdays & weekend

```
newcodingdat <- log_fires
newcodingdat$weekend <- ifelse(newcodingdat$day == "sat" |
                              newcodingdat$day == "sun", 1, 0)
fires_lm = newcodingdat[,-c(3,4)]
#head(fires_lm)
#fitting the most generic model
linear_fit = lm(log_area~., data = fires_lm[-test,])
summary(linear_fit)
```

```
##
## Call:
## lm(formula = log_area ~ ., data = fires_lm[-test, ])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.6752 -1.1002 -0.5809  0.9121  5.4788
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.3203213  1.4970338  -0.214   0.8307
## X            0.0444227  0.0366203   1.213   0.2258
## Y            0.0186167  0.0687691   0.271   0.7868
## FFMC         0.0083989  0.0155467   0.540   0.5893
## DMC          0.0009770  0.0017170   0.569   0.5697
## DC           0.0002326  0.0004121   0.564   0.5729
## ISI         -0.0256429  0.0183791  -1.395   0.1637
## temp         0.0132434  0.0201772   0.656   0.5120
## RH          -0.0038001  0.0060715  -0.626   0.5317
## wind         0.0750019  0.0423335   1.772   0.0772 .
## rain         0.0908921  0.2205852   0.412   0.6805
## weekend      0.1161755  0.1528106   0.760   0.4475
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.431 on 402 degrees of freedom
## Multiple R-squared:  0.02815,    Adjusted R-squared:  0.001555
## F-statistic: 1.058 on 11 and 402 DF,  p-value: 0.394
```

```
linear_fit = glm(log_area ~ ., data = fires_lm[-test,])

#10 CV validation for the training set
cv.glm(fires_lm[-test,], linear_fit, K = 10)$delta[1]
```

```
## [1] 2.21769
```

```
# getting MSE for the test
pred = predict(linear_fit, fires_lm[test,])
mean((pred - fires_lm[test, "log_area"])**2)
```

```
## [1] 1.567718
```

coding seasons too

```
newcodingdat$season <- newcodingdat$month
newcodingdat$season<- recode(newcodingdat$season, jan = "winter", feb = "winter",
       mar = "spring", apr = "spring", may = "spring",
       jun = "summer", jul = "summer", aug = "summer",
       sep = "fall", oct = "fall", nov = "fall",
       dec = "winter")
newdata_2 = one_hot(as.data.table(newcodingdat[, -c(3,4)]))
fires_lm = newdata_2
linear_fit = lm(log_area~., data = newdata_2[-test,])
summary(linear_fit) # this one seem to be better(?)
```

```
##
## Call:
## lm(formula = log_area ~ ., data = newdata_2[-test, ])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9740 -1.0766 -0.5295  0.9088  5.2195
##
## Coefficients: (1 not defined because of singularities)
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -1.3378952  1.6758055  -0.798   0.4251
## X               0.0469752  0.0364158   1.290   0.1978
## Y               0.0086438  0.0685471   0.126   0.8997
## FFMC            0.0146339  0.0158625   0.923   0.3568
## DMC             0.0027742  0.0019621   1.414   0.1582
## DC             -0.0003841  0.0008627  -0.445   0.6564
## ISI            -0.0192240  0.0186576  -1.030   0.3035
## temp            0.0405492  0.0237187   1.710   0.0881 .
## RH             0.0010194  0.0065173   0.156   0.8758
## wind           0.0887496  0.0425488   2.086   0.0376 *
## rain           0.0636258  0.2205348   0.289   0.7731
## weekend        0.1234383  0.1519818   0.812   0.4172
## season_spring -0.2697042  0.5581697  -0.483   0.6292
## season_summer -0.4857303  0.2314794  -2.098   0.0365 *
## season_winter  0.4281127  0.5709725   0.750   0.4538
## season_fall           NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.422 on 399 degrees of freedom
## Multiple R-squared:  0.04774,    Adjusted R-squared:  0.01433
## F-statistic: 1.429 on 14 and 399 DF,  p-value: 0.1362
```

```
linear_fit = glm(log_area ~ ., data = fires_lm[-test,])

#10 CV validation for the training set
cv.glm(fires_lm[-test,], linear_fit, K = 10)$delta[1]
```

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

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

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

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

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

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

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

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

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

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

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

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
```

```
## prediction from a rank-deficient fit may be misleading

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

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

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

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

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

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

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

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

## [1] 2.21272
```

```r
# getting MSE for the test
pred = predict(linear_fit, fires_lm[test,])
```

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

```r
sum((pred - fires_lm[test, "log_area"])**2)/103
```

```
## [1] 1.536772
```

```r
mean((pred - fires_lm[test, "log_area"])**2)
```

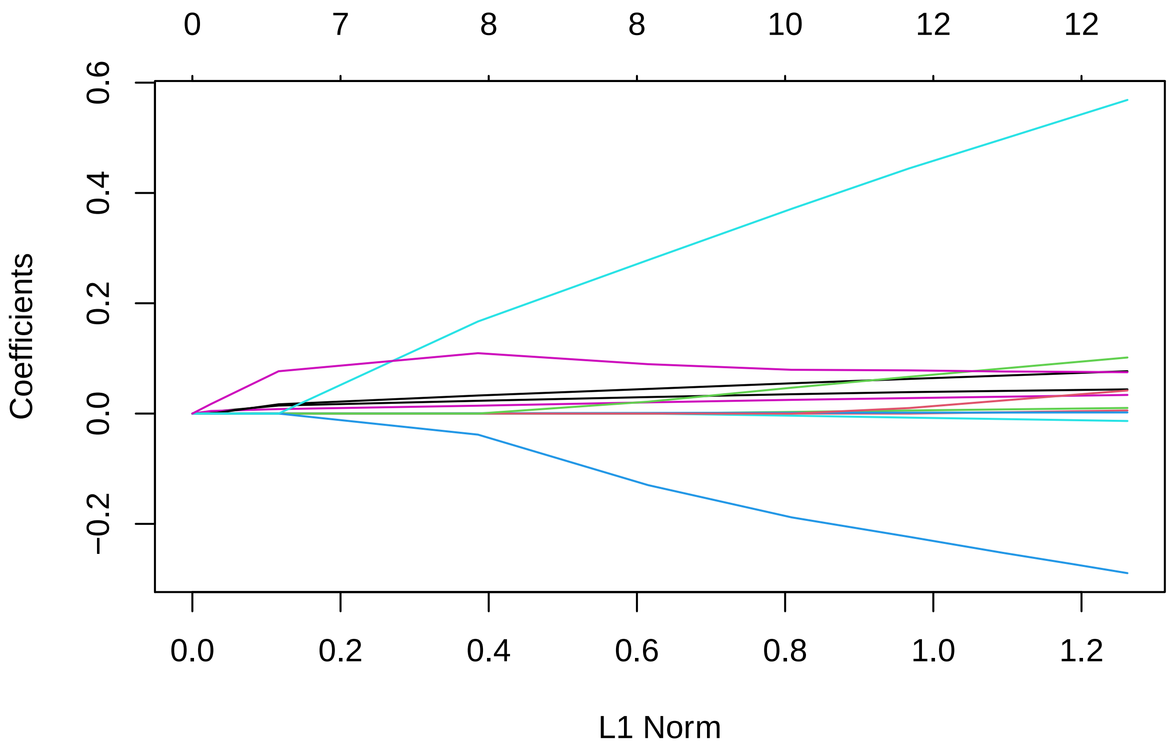
```
## Warning in mean.default((pred - fires_lm[test, "log_area"])^2): argument is not
## numeric or logical: returning NA

## [1] NA
```

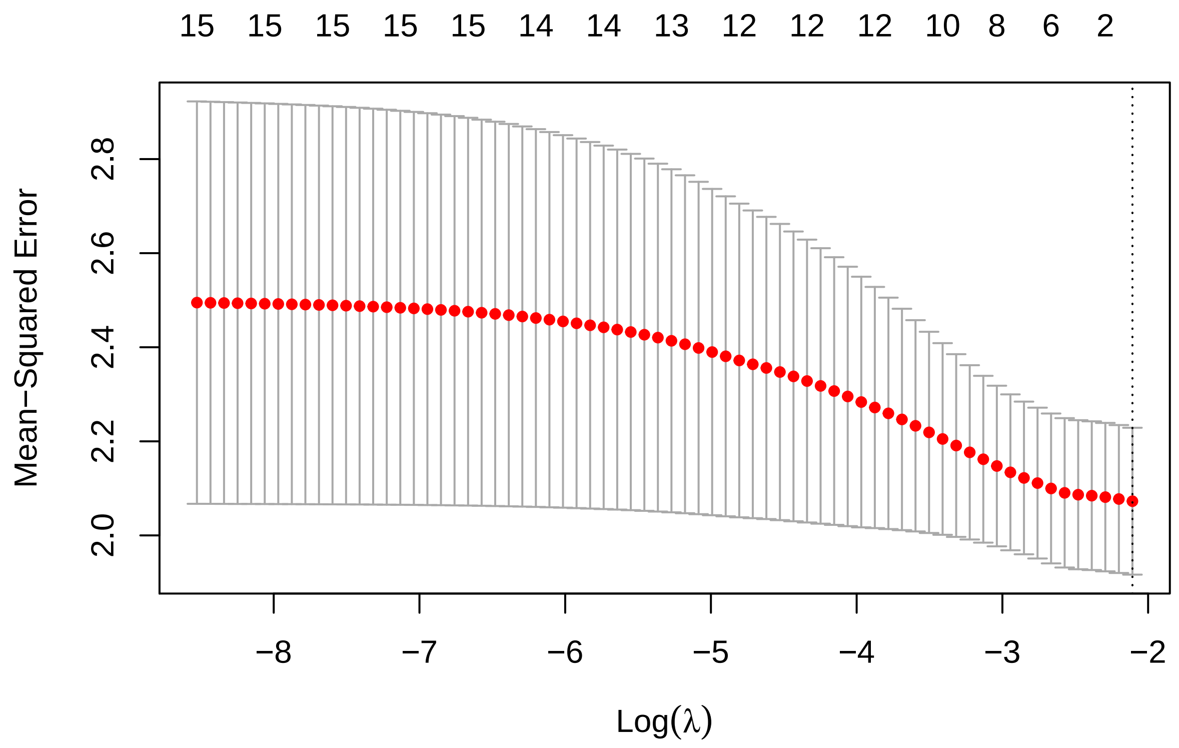
Lasso after new coding

```r
grid=10^seq(10,-2,length=100)
lasso_mod = glmnet(x = as.matrix(newdata_2[-test, -"log_area"]), y = as.matrix(newdata_2[-test, log_area
plot(lasso_mod)
```

```
## Warning in regularize.values(x, y, ties, missing(ties), na.rm = na.rm):
## collapsing to unique 'x' values
```



```
# cross validation
cv_lasso = cv.glmnet(x = as.matrix(newdata_2[-test, -"log_area"]), y = as.matrix(newdata_2[-test, log_a
plot(cv_lasso)
```



```
bestlam = cv_lasso$lambda.min
lasso_pred=predict(lasso_mod,s=bestlam, newx=as.matrix(newdata_2[-test, -"log_area"]))
mean((lasso_pred-as.matrix(newdata_2[-test, log_area]))^2 - as.matrix(newdata_2[-test, log_area]))
```

```
## [1] 0.8884434
```

```
out=glmnet(as.matrix(newdata_2[, -"log_area"]), as.matrix(newdata_2[, log_area]), alpha=1,lambda=grid)
lasso_coef=predict(out,type="coefficients",s=bestlam)
lasso_coef
```

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                           1
## (Intercept)    1.110803e+00
## X              .
## Y              .
## FFMC           .
## DMC            .
## DC             .
## ISI            .
## temp           .
## RH             .
## wind           6.058556e-05
## rain           .
## weekend        .
## season_spring -1.005715e-03
## season_summer  .
## season_winter  .
## season_fall    2.900494e-04
```

Classification

```
median(log_fires$log_area) #values below this are considered small fires
```

```
## [1] 0.4187103
```

```
bi_fires <- log_fires
bi_fires$large_fire <- ifelse(bi_fires$log_area < median(log_fires$log_area), 0, 1)
fires_lm <- bi_fires[, -c(1,2,3,4,13)]

# Logit
set.seed(1)
test = sample(nrow(fires_lm), floor(nrow(fires_lm)*.2))

# train the model on training set
train_control <- trainControl(method = "cv", number = 10)

logit_fit <- train(as.factor(large_fire) ~ .,
               data = fires_lm[-test, ],
               trControl = train_control,
               method = "glm",
               family=binomial())
summary(logit_fit)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.4727  -1.2106   0.9994   1.1235   1.4294
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.0818105  2.4331596  -1.267    0.205
## FFMC         0.0252088  0.0263167   0.958    0.338
## DMC         -0.0006551  0.0024048  -0.272    0.785
## DC           0.0005340  0.0005731   0.932    0.351
## ISI         -0.0207287  0.0273417  -0.758    0.448
## temp         0.0246392  0.0280009   0.880    0.379
## RH           0.0010634  0.0084145   0.126    0.899
## wind         0.0813731  0.0602095   1.351    0.177
## rain         0.2304277  0.4371396   0.527    0.598
##
## (Dispersion parameter for binomial family taken to be 1)
##
```

```
##     Null deviance: 573.31  on 413  degrees of freedom
## Residual deviance: 565.61  on 405  degrees of freedom
## AIC: 583.61
##
## Number of Fisher Scoring iterations: 4
```

```
prob = predict(logit_fit, fires_lm[test,], type = "prob")
pred=rep(0, 517)
pred[prob >0.5]=1
table(pred,as.factor(fires_lm$large_fire))
```

```
##
## pred   0   1
##    0 153 130
##    1 104 130
```

Adding location variable

```
fires_lm <- bi_fires[, -c(3,4,13)]

# Logit
set.seed(1)
test = sample(nrow(fires_lm), floor(nrow(fires_lm)*.2))

# train the model on training set
train_control <- trainControl(method = "cv", number = 10)

logit_fit <- train(as.factor(large_fire) ~ .,
               data = fires_lm[-test, ],
               trControl = train_control,
               method = "glm",
               family=binomial())
summary(logit_fit)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5118  -1.1892   0.9343   1.1192   1.4356
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.506e+00  2.488e+00  -1.409    0.159
## X            5.880e-02  5.173e-02   1.137    0.256
## Y            4.869e-02  9.737e-02   0.500    0.617
## FFMC         2.474e-02  2.657e-02   0.931    0.352
## DMC         -6.617e-04  2.437e-03  -0.272    0.786
## DC           6.431e-04  5.819e-04   1.105    0.269
## ISI         -2.114e-02  2.732e-02  -0.774    0.439
## temp         2.248e-02  2.810e-02   0.800    0.424
## RH          -2.552e-05  8.450e-03  -0.003    0.998
## wind         8.505e-02  6.052e-02   1.405    0.160
## rain         1.998e-01  4.305e-01   0.464    0.643
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 573.31  on 413  degrees of freedom
## Residual deviance: 562.63  on 403  degrees of freedom
## AIC: 584.63
##
## Number of Fisher Scoring iterations: 4
```

```
prob = predict(logit_fit, fires_lm[test,], type = "prob")
pred=rep(0, 517)
pred[prob >0.5]=1
table(pred,as.factor(fires_lm$large_fire))
```

```
##
## pred   0   1
##    0 141 134
##    1 116 126
```

All variables

```
fires_lm <- one_hot(as.data.table(bi_fires))
fires_lm <- fires_lm[, -c(30)]
# Logit
set.seed(1)
test = sample(nrow(fires_lm), floor(nrow(fires_lm)*.2))

# train the model on training set
train_control <- trainControl(method = "cv", number = 10)

logit_fit <- train(as.factor(large_fire) ~ .,
               data = fires_lm[-test, ],
               trControl = train_control,
               method = "glm",
               family=binomial())
```

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

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

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

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

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
```

```
## prediction from a rank-deficient fit may be misleading

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

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

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

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

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

```r
summary(logit_fit)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7771  -1.1606   0.7457   1.1188   1.6854
##
## Coefficients: (2 not defined because of singularities)
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -4.990e+00  3.493e+00  -1.429    0.153
## X            6.502e-02  5.404e-02   1.203    0.229
## Y            3.848e-03  1.013e-01   0.038    0.970
## month_apr    1.326e+00  1.522e+00   0.871    0.384
## month_aug   -1.247e-01  3.730e-01  -0.334    0.738
## month_dec    1.637e+01  5.779e+02   0.028    0.977
## month_feb    8.972e-01  1.365e+00   0.657    0.511
## month_jan   -1.192e+01  1.455e+03  -0.008    0.993
## month_jul    1.596e-01  7.174e-01   0.223    0.824
## month_jun    5.858e-01  1.070e+00   0.548    0.584
## month_mar    2.643e-01  1.281e+00   0.206    0.836
## month_may    5.752e-01  1.874e+00   0.307    0.759
## month_nov   -1.465e+01  1.455e+03  -0.010    0.992
## month_oct   -3.807e-01  6.826e-01  -0.558    0.577
## month_sep           NA         NA      NA       NA
## day_fri     -4.827e-01  4.205e-01  -1.148    0.251
## day_mon     -2.565e-01  4.346e-01  -0.590    0.555
## day_sat     -1.349e-01  4.215e-01  -0.320    0.749
## day_sun     -5.799e-01  4.101e-01  -1.414    0.157
## day_thu     -6.758e-01  4.408e-01  -1.533    0.125
## day_tue     -2.213e-01  4.415e-01  -0.501    0.616
## day_wed             NA         NA      NA       NA
## FFMC         3.439e-02  3.309e-02   1.039    0.299
## DMC         -3.384e-04  3.125e-03  -0.108    0.914
## DC           1.205e-03  2.054e-03   0.586    0.558
## ISI         -1.857e-02  2.959e-02  -0.628    0.530
## temp         4.858e-02  3.704e-02   1.311    0.190
## RH           4.711e-03  1.037e-02   0.455    0.649
## wind         7.495e-02  6.470e-02   1.158    0.247
## rain         1.568e-01  4.460e-01   0.352    0.725
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 573.31  on 413  degrees of freedom
## Residual deviance: 543.88  on 386  degrees of freedom
## AIC: 599.88
##
## Number of Fisher Scoring iterations: 14
```

```
prob = predict(logit_fit, fires_lm[test,], type = "prob")
```

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

```
pred=rep(0, 517)
pred[prob >0.5]=1
table(pred,as.factor(fires_lm$large_fire))
```

```
##
## pred   0   1
##    0 148 127
##    1 109 133
```

Using weekday/weekend and seasons

```
fires_lm = newdata_2
fires_lm$large_fire = ifelse(fires_lm$log_area < median(fires_lm$log_area), 0, 1)
fires_lm = fires_lm[, -c(16)]

set.seed(1)
test = sample(nrow(fires_lm), floor(nrow(fires_lm)*.2))

# train the model on training set
train_control <- trainControl(method = "cv", number = 10)

logit_fit <- train(as.factor(large_fire) ~ .,
               data = fires_lm[-test, ],
               trControl = train_control,
               method = "glm",
               family=binomial())
```

```
## Warning: glm.fit: algorithm did not converge
```

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

```
## Warning: glm.fit: algorithm did not converge
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

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

## Warning: glm.fit: algorithm did not converge

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

## Warning: glm.fit: algorithm did not converge

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

## Warning: glm.fit: algorithm did not converge

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

## Warning: glm.fit: algorithm did not converge

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

## Warning: glm.fit: algorithm did not converge

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

## Warning: glm.fit: algorithm did not converge

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

## Warning: glm.fit: algorithm did not converge

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

## Warning: glm.fit: algorithm did not converge

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

```
summary(logit_fit)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##        Min          1Q      Median          3Q         Max
## -1.925e-04  -2.100e-08   2.100e-08   2.100e-08   1.480e-04
##
## Coefficients:
```

```
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)    7.008e+01  1.923e+05   0.000    1.000
## X              6.514e+00  5.028e+03   0.001    0.999
## Y             -1.307e+01  1.191e+04  -0.001    0.999
## FFMC          -1.588e+00  8.846e+02  -0.002    0.999
## DMC           -3.828e-02  4.543e+02   0.000    1.000
## DC            -8.733e-03  1.465e+02   0.000    1.000
## ISI            5.636e-01  1.274e+03   0.000    1.000
## temp           2.552e+00  2.805e+03   0.001    0.999
## RH            -1.317e-01  8.079e+02   0.000    1.000
## wind          -2.297e-01  3.013e+03   0.000    1.000
## rain          -5.064e+01  4.246e+04  -0.001    0.999
## log_area       1.619e+02  2.010e+04   0.008    0.994
## weekend       -3.964e+00  1.729e+04   0.000    1.000
## season_spring -2.029e+01  1.205e+05   0.000    1.000
## season_summer -3.170e+00  4.046e+04   0.000    1.000
## season_winter -1.647e+01  8.981e+04   0.000    1.000
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5.7331e+02  on 413  degrees of freedom
## Residual deviance: 1.3964e-07  on 398  degrees of freedom
## AIC: 32
##
## Number of Fisher Scoring iterations: 25
```

```
prob = predict(logit_fit, fires_lm[test,], type = "prob")
pred=rep(0, 517)
pred[prob >0.5]=1
table(pred,as.factor(fires_lm$large_fire))
```

```
##
## pred   0   1
##    0 113 141
##    1 144 119
```

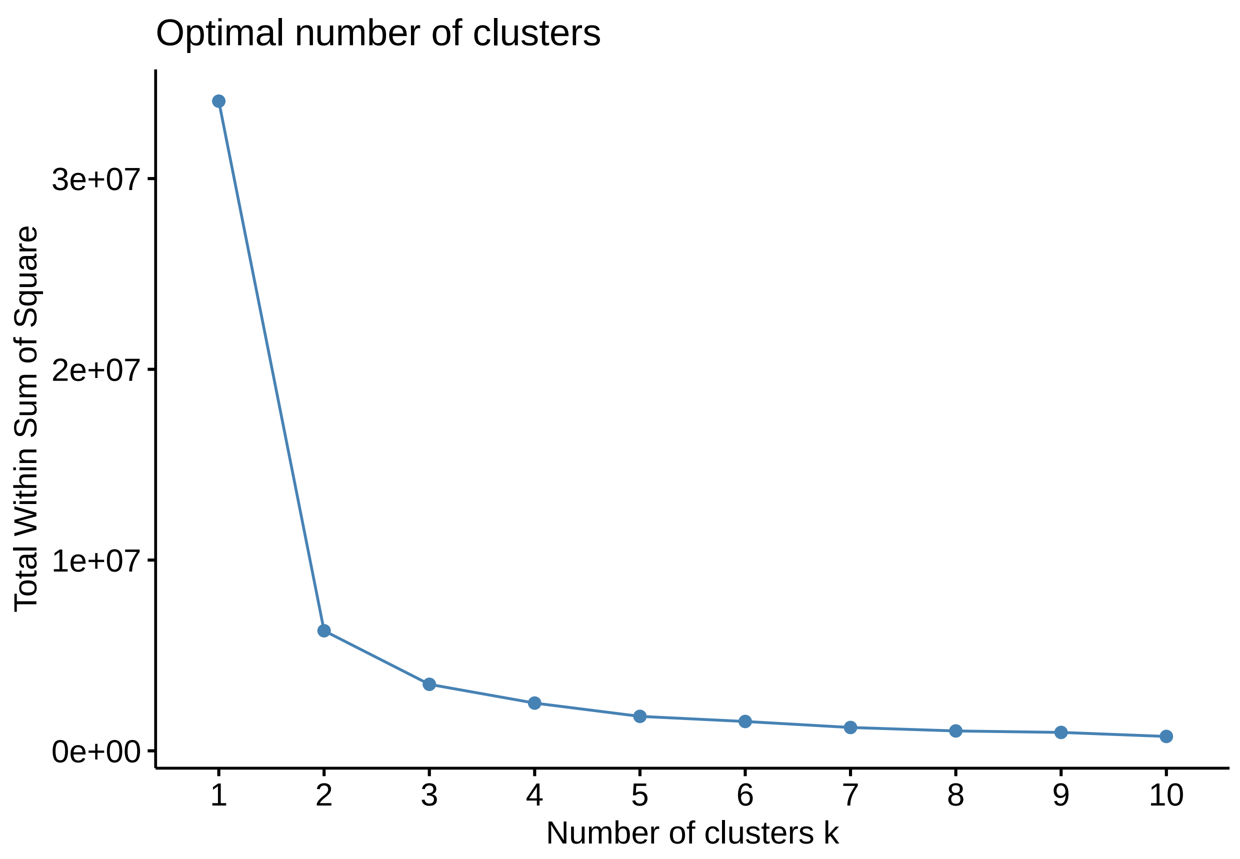
Clustering methods

k-medoid clustering

```
pam(newdata, 2, metric = "euclidean", stand = TRUE)
```

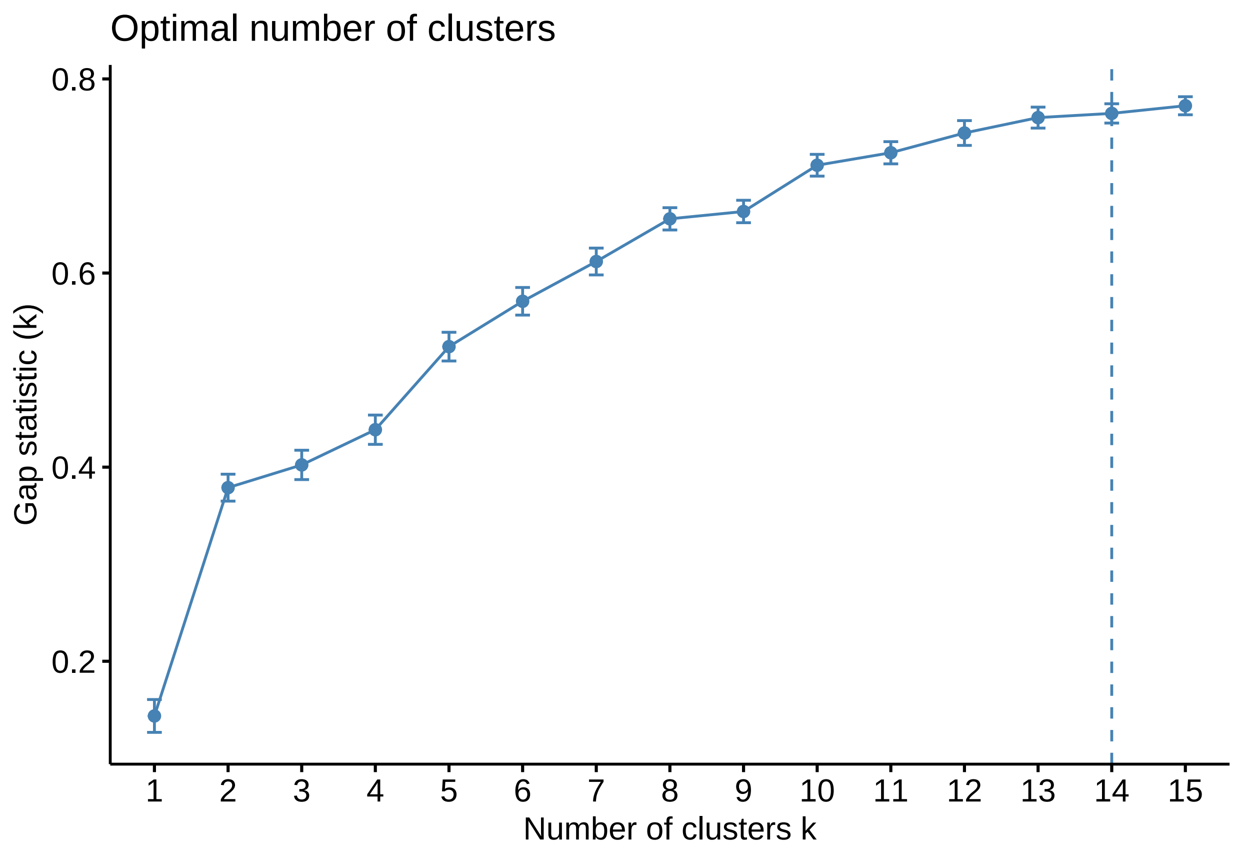
```
## Medoids:
##       ID X Y month_apr month_aug month_dec month_feb month_jan month_jul
## [1,] 357 4 4         0         0         0         0         0         0
## [2,] 101 3 4         0         1         0         0         0         0
##      month_jun month_mar month_may month_nov month_oct month_sep day_fri
## [1,]         0         0         0         0         0         1       1
## [2,]         0         0         0         0         0         0       0
##      day_mon day_sat day_sun day_thu day_tue day_wed FFMC   DMC    DC  ISI temp
## [1,]       0       0       0       0       0       0 92.1  99.0 745.3  9.6 20.8
## [2,]       0       0       1       0       0       0 91.4 142.4 601.4 10.6 19.8
##      RH wind rain  log_area
## [1,] 35  4.9    0 0.8153648
## [2,] 39  5.4    0 0.0000000
## Clustering vector:
##   [1] 1 1 1 1 2 2 2 2 1 1 1 1 1 1 1 1 2 1 2 2 1 1 2 2 2 2 1 1 1 2 1 1 1 2 1 1 1
##  [38] 1 1 2 2 2 2 1 1 1 1 1 2 2 1 2 2 2 2 1 1 2 2 1 2 2 2 2 2 2 1 1 1 1 1 1 1 2
##  [75] 1 2 1 2 1 2 2 2 2 2 2 2 1 1 1 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 1 2 1 2 2 1 1 1
## [112] 1 2 1 2 2 2 2 2 2 2 2 2 1 1 2 2 1 1 2 2 2 2 1 2 2 2 1 1 1 1 2 1 2 2 2 2 1
## [149] 2 1 1 2 1 1 1 2 1 2 2 1 1 2 2 1 2 2 2 2 1 2 1 2 2 1 2 2 2 2 1 2 2 1 2 1 1
## [186] 1 2 1 1 2 1 2 2 1 2 1 2 1 1 1 1 2 2 1 2 1 2 1 1 1 1 2 1 2 1 1 2 1 1 1 2 1
## [223] 2 1 1 2 1 2 2 2 1 2 1 1 1 2 1 1 1 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [260] 2 2 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 1 1 2 2 1 2 2 2 1 1 2 1 1 1 1 2
## [297] 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [334] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2
## [371] 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 2 2 2 1 1 2 1 2 2 2 2 2 2 2 1 1 1 1 2 1 2 1
## [408] 2 1 2 1 1 1 2 2 2 1 2 2 2 2 2 2 1 2 2 2 2 2 2 1 2 2 2 1 2 2 2 2 1 1 2 2 1
## [445] 1 2 2 2 1 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 1 2 1 1 2 1 2 2 1 1
## [482] 2 2 2 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 2 2 2 2 2 2
## Objective function:
##    build     swap
## 10.34618 10.34618
##
## Available components:
##  [1] "medoids"    "id.med"     "clustering" "objective"  "isolation"
##  [6] "clusinfo"   "silinfo"    "diss"       "call"       "data"
```

```
fviz_nbclust(newdata, pam, method = "wss")
```



```
# look at elbow --> three clusters

# another approach
gap_stat <- clusGap(newdata, FUN = pam, K.max = 15,B = 50)
fviz_gap_stat(gap_stat)
```

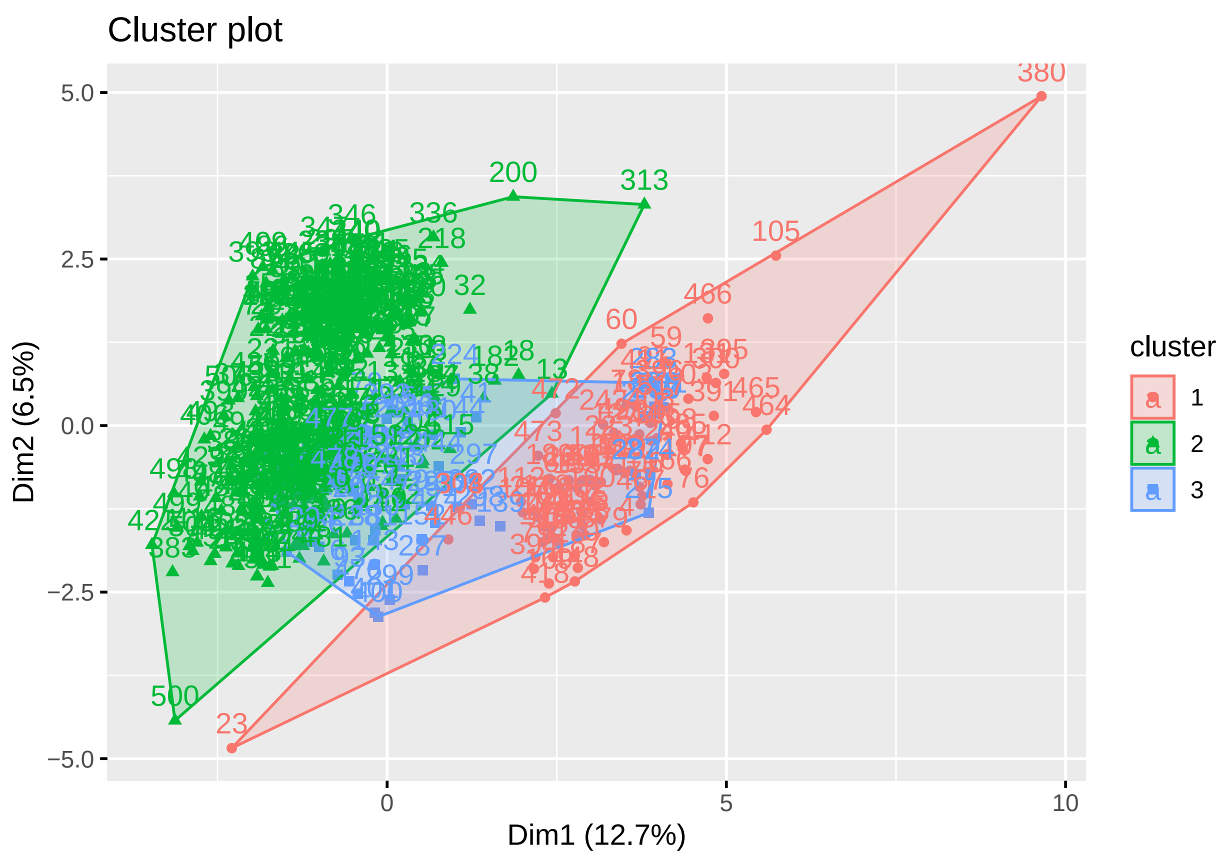


```r
# k =14 -> no, overfit

# choosing 3 clusters seem fine?
set.seed(1)
kmed <- pam(newdata, k = 3)
kmed
```

```
## Medoids:
##       ID X Y month_apr month_aug month_dec month_feb month_jan month_jul
## [1,]  40 4 4         0         0         0         0         0         0
## [2,]  14 6 5         0         0         0         0         0         0
## [3,] 478 4 3         0         0         0         0         0         1
##      month_jun month_mar month_may month_nov month_oct month_sep day_fri
## [1,]         0         1         0         0         0         0       0
## [2,]         0         0         0         0         0         1       0
## [3,]         0         0         0         0         0         0       0
##      day_mon day_sat day_sun day_thu day_tue day_wed FFMC   DMC    DC  ISI temp
## [1,]       0       0       0       0       1       0 88.1  25.7  67.6  3.8 14.1
## [2,]       1       0       0       0       0       0 90.9 126.5 686.5  7.0 21.3
## [3,]       0       0       1       0       0       0 93.7 101.3 423.4 14.7 26.1
##      RH wind rain log_area
## [1,] 43  2.7    0 0.000000
## [2,] 42  2.2    0 0.000000
## [3,] 45  4.0    0 2.123458
## Clustering vector:
##   [1] 1 2 2 1 1 3 3 2 2 2 2 2 2 2 2 2 1 2 1 1 2 2 1 3 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [38] 2 2 1 3 2 2 2 2 2 2 3 1 1 2 2 2 2 2 2 2 2 1 1 1 1 3 2 2 2 2 2 2 1 1 2 1 2
##  [75] 2 1 1 1 3 2 2 2 2 2 2 2 2 2 2 1 2 1 3 2 2 2 1 1 2 2 2 2 2 2 1 1 1 2 2 2 1
## [112] 1 2 2 1 1 1 1 1 3 2 2 2 2 2 2 1 2 2 2 1 1 1 2 1 2 2 2 3 2 2 3 3 3 3 2 1 2
## [149] 2 2 3 3 3 2 2 2 2 2 2 2 2 1 2 1 2 2 1 2 2 1 2 2 2 2 2 2 2 1 2 2 2 2 2 1 2 2
## [186] 2 1 2 1 1 1 1 2 2 2 2 2 2 1 2 2 2 2 1 1 2 1 2 2 2 2 2 2 2 2 1 1 1 2 2 2 1 2 2
## [223] 1 3 2 2 2 3 2 3 2 2 2 2 2 2 2 2 2 1 1 1 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2
## [260] 2 3 2 2 2 2 3 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 1 1 3 3 3 3 3 3 3 3 3 3 3
## [297] 3 3 3 1 3 3 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [334] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [371] 2 3 2 2 2 2 2 2 1 1 3 2 2 2 2 2 2 1 2 2 1 2 2 1 1 1 2 2 2 3 3 2 2 2 2 2 2
## [408] 1 2 3 1 1 3 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 1 3
## [445] 2 1 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 3 3 3 3 3 3 3 2
## [482] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1
## Objective function:
##    build     swap
## 77.75335 71.39347
##
## Available components:
##  [1] "medoids"    "id.med"     "clustering" "objective"  "isolation"
##  [6] "clusinfo"   "silinfo"    "diss"       "call"       "data"
```

```r
fviz_cluster(kmed, data = newdata)
```



```r
# combine clusters into the dataset
labeled_data = cbind(newdata, cluster = kmed$cluster)
head(labeled_data)
```

```
##    X Y month_apr month_aug month_dec month_feb month_jan month_jul month_jun
## 1: 7 5         0         0         0         0         0         0         0
## 2: 7 4         0         0         0         0         0         0         0
## 3: 7 4         0         0         0         0         0         0         0
## 4: 8 6         0         0         0         0         0         0         0
## 5: 8 6         0         0         0         0         0         0         0
## 6: 8 6         0         1         0         0         0         0         0
##    month_mar month_may month_nov month_oct month_sep day_fri day_mon day_sat
## 1:         1         0         0         0         0       1       0       0
## 2:         0         0         0         1         0       0       0       0
## 3:         0         0         0         1         0       0       0       1
## 4:         1         0         0         0         0       1       0       0
## 5:         1         0         0         0         0       0       0       0
## 6:         0         0         0         0         0       0       0       0
##    day_sun day_thu day_tue day_wed FFMC  DMC    DC  ISI temp RH wind rain
## 1:       0       0       0       0 86.2 26.2  94.3  5.1  8.2 51  6.7  0.0
## 2:       0       0       1       0 90.6 35.4 669.1  6.7 18.0 33  0.9  0.0
## 3:       0       0       0       0 90.6 43.7 686.9  6.7 14.6 33  1.3  0.0
## 4:       0       0       0       0 91.7 33.3  77.5  9.0  8.3 97  4.0  0.2
## 5:       1       0       0       0 89.3 51.3 102.2  9.6 11.4 99  1.8  0.0
## 6:       1       0       0       0 92.3 85.3 488.0 14.7 22.2 29  5.4  0.0
##    log_area cluster
## 1:        0       1
## 2:        0       2
## 3:        0       2
## 4:        0       1
## 5:        0       1
## 6:        0       3
```

```
summary(lm(log_area ~ as.factor(cluster), data = labeled_data))
```

```
##
## Call:
## lm(formula = log_area ~ as.factor(cluster), data = labeled_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.2338 -1.1595 -0.7576  0.8985  5.8362
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)           0.8391     0.1447   5.800 1.16e-08 ***
## as.factor(cluster)2   0.3204     0.1623   1.974   0.0489 *
## as.factor(cluster)3   0.3947     0.2266   1.742   0.0821 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.395 on 514 degrees of freedom
## Multiple R-squared:  0.00861,    Adjusted R-squared:  0.004752
## F-statistic: 2.232 on 2 and 514 DF,  p-value: 0.1084
```

```
# significant....
```

```r
# Look into these clusters
mod_cl1 <-lm(log_area ~ ., data = labeled_data[cluster == 1,-c(31)])
summary(mod_cl1)
```

```
##
## Call:
## lm(formula = log_area ~ ., data = labeled_data[cluster == 1,
##     -c(31)])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.6170 -0.7117 -0.1905  0.3952  3.0386
##
## Coefficients: (6 not defined because of singularities)
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.880739   3.815791   0.493  0.62366
## X            0.279519   0.096421   2.899  0.00502 **
## Y           -0.240244   0.132721  -1.810  0.07463 .
## month_apr    0.692838   1.637681   0.423  0.67357
## month_aug    0.590740   2.237398   0.264  0.79254
## month_dec          NA         NA      NA       NA
## month_feb    1.293663   1.614711   0.801  0.42578
## month_jan    1.366763   1.974383   0.692  0.49110
## month_jul          NA         NA      NA       NA
## month_jun    0.716238   2.013268   0.356  0.72311
## month_mar    0.571056   1.648783   0.346  0.73013
## month_may    2.733079   1.829863   1.494  0.13984
## month_nov          NA         NA      NA       NA
## month_oct          NA         NA      NA       NA
## month_sep          NA         NA      NA       NA
## day_fri      0.202898   0.625582   0.324  0.74667
## day_mon      0.556858   0.626098   0.889  0.37687
## day_sat      0.562707   0.617611   0.911  0.36541
## day_sun      1.162747   0.672619   1.729  0.08834 .
## day_thu      0.653469   0.693031   0.943  0.34901
## day_tue      0.306345   0.705204   0.434  0.66535
## day_wed            NA         NA      NA       NA
## FFMC        -0.006191   0.030231  -0.205  0.83834
## DMC          0.032623   0.027364   1.192  0.23728
## DC          -0.010353   0.011240  -0.921  0.36023
## ISI         -0.043229   0.030278  -1.428  0.15788
## temp        -0.082702   0.062433  -1.325  0.18966
## RH          -0.025600   0.014577  -1.756  0.08350 .
## wind         0.038346   0.081678   0.469  0.64021
## rain         1.198208   7.224578   0.166  0.86876
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.205 on 69 degrees of freedom
## Multiple R-squared:  0.2867, Adjusted R-squared:  0.04889
## F-statistic: 1.206 on 23 and 69 DF,  p-value: 0.2706
```

```r
mod_cl2 <-lm(log_area ~ ., data = labeled_data[cluster == 2,-c(31)])
summary(mod_cl2)
```

```
##
## Call:
## lm(formula = log_area ~ ., data = labeled_data[cluster == 2,
##     -c(31)])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9872 -1.0845 -0.4412  0.8243  5.2307
##
## Coefficients: (10 not defined because of singularities)
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.8953488  2.9450979   0.644   0.5203
## X           -0.0102359  0.0389513  -0.263   0.7929
## Y            0.1185907  0.0787965   1.505   0.1332
## month_apr           NA         NA      NA       NA
## month_aug   -0.7109636  0.2737174  -2.597   0.0098 **
## month_dec           NA         NA      NA       NA
## month_feb           NA         NA      NA       NA
## month_jan           NA         NA      NA       NA
## month_jul   -1.0136358  0.7295056  -1.389   0.1656
## month_jun           NA         NA      NA       NA
## month_mar           NA         NA      NA       NA
## month_may           NA         NA      NA       NA
## month_nov           NA         NA      NA       NA
## month_oct   -0.1712450  0.4212835  -0.406   0.6846
## month_sep           NA         NA      NA       NA
## day_fri     -0.2367889  0.3089932  -0.766   0.4440
## day_mon     -0.4160314  0.3227245  -1.289   0.1982
## day_sat     -0.0113225  0.3057996  -0.037   0.9705
## day_sun     -0.1712231  0.3031773  -0.565   0.5726
## day_thu     -0.3051454  0.3200561  -0.953   0.3411
## day_tue      0.0871540  0.3167490   0.275   0.7834
## day_wed             NA         NA      NA       NA
## FFMC         0.0014381  0.0265253   0.054   0.9568
## DMC          0.0040055  0.0021941   1.826   0.0688 .
## DC          -0.0029709  0.0020109  -1.477   0.1405
## ISI         -0.0221861  0.0267755  -0.829   0.4079
## temp         0.0350822  0.0288084   1.218   0.2242
## RH          -0.0005991  0.0090897  -0.066   0.9475
## wind         0.0581548  0.0524788   1.108   0.2686
## rain         0.0541111  0.2255372   0.240   0.8105
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.422 on 340 degrees of freedom
## Multiple R-squared:  0.05797,    Adjusted R-squared:  0.005327
## F-statistic: 1.101 on 19 and 340 DF,  p-value: 0.3476
```

```r
mod_cl3 <-lm(log_area ~ ., data = labeled_data[cluster == 3,-c(31)])
summary(mod_cl3)
```

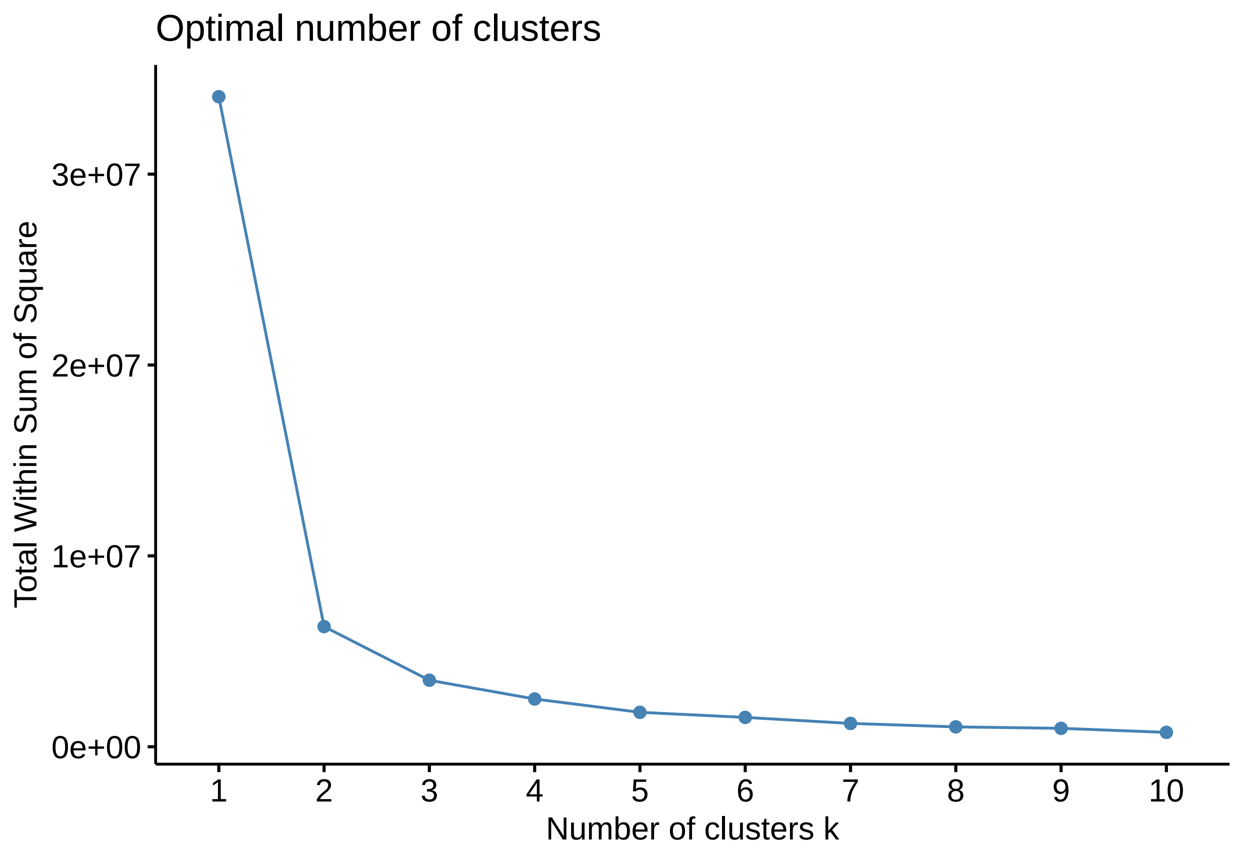
```
##
## Call:
## lm(formula = log_area ~ ., data = labeled_data[cluster == 3,
##     -c(31)])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.7902 -0.7924 -0.0436  0.5683  3.3758
##
## Coefficients: (9 not defined because of singularities)
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -15.747076  10.545432  -1.493 0.142671
## X             0.216761   0.087966   2.464 0.017809 *
## Y            -0.158052   0.155313  -1.018 0.314545
## month_apr           NA         NA      NA       NA
## month_aug     2.167829   0.963156   2.251 0.029569 *
## month_dec     5.467049   1.474101   3.709 0.000592 ***
## month_feb     3.217327   1.938548   1.660 0.104260
## month_jan           NA         NA      NA       NA
## month_jul     1.105870   0.663726   1.666 0.102952
## month_jun           NA         NA      NA       NA
## month_mar           NA         NA      NA       NA
## month_may           NA         NA      NA       NA
## month_nov           NA         NA      NA       NA
## month_oct           NA         NA      NA       NA
## month_sep           NA         NA      NA       NA
## day_fri       0.608689   0.707888   0.860 0.394631
## day_mon       0.894407   0.659843   1.355 0.182341
## day_sat       0.814813   0.634468   1.284 0.205932
## day_sun      -0.466837   0.638672  -0.731 0.468775
## day_thu       0.175287   0.723678   0.242 0.809763
## day_tue       0.864059   0.756589   1.142 0.259755
## day_wed             NA         NA      NA       NA
## FFMC          0.167334   0.118671   1.410 0.165714
## DMC           0.015513   0.007117   2.180 0.034791 *
## DC           -0.012736   0.004531  -2.811 0.007406 **
## ISI          -0.037732   0.095573  -0.395 0.694944
## temp          0.104221   0.064940   1.605 0.115838
## RH            0.028292   0.016754   1.689 0.098528 .
## wind          0.071672   0.121909   0.588 0.559669
## rain        -11.935854   8.337375  -1.432 0.159485
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.284 on 43 degrees of freedom
## Multiple R-squared:  0.4526, Adjusted R-squared:  0.1981
## F-statistic: 1.778 on 20 and 43 DF,  p-value: 0.05688
```

Try without qualitative variables?

```r
clusterdat <- as.data.frame(log_fires[, -c(3,4)])
pam(clusterdat, 2, metric = "euclidean", stand = TRUE)
```

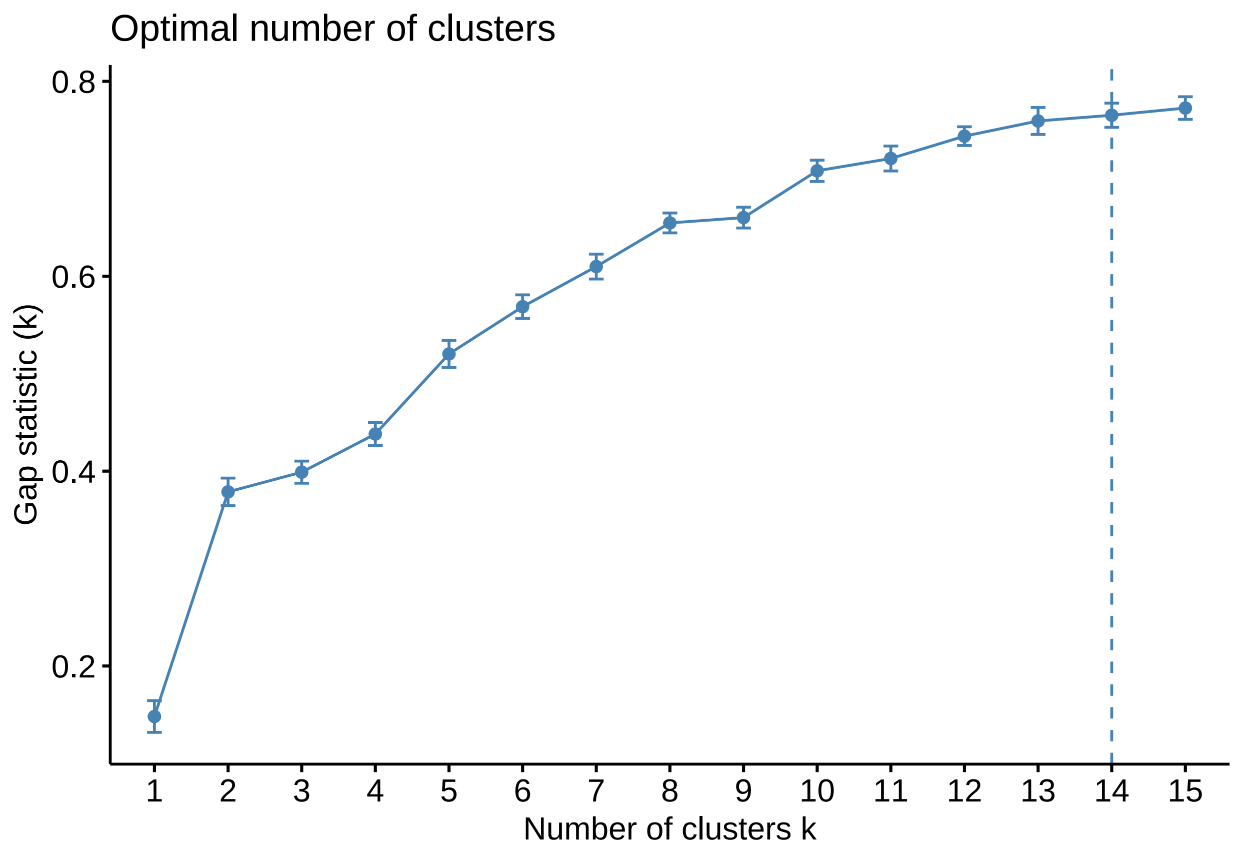
```
## Medoids:
##       ID X Y FFMC  DMC    DC ISI temp RH wind rain  log_area
## [1,] 147 5 4 90.1 39.7  86.6 6.2 13.2 40  5.4    0 0.6678294
## [2,] 357 4 4 92.1 99.0 745.3 9.6 20.8 35  4.9    0 0.8153648
## Clustering vector:
##   [1] 1 2 2 1 1 2 2 2 2 2 2 2 1 2 2 2 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2
##  [38] 1 2 1 1 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 1 1 1 1 2 2 2 2 2 2 2 1 1 2 1 2
##  [75] 2 1 1 1 2 2 2 2 2 2 2 2 2 2 2 1 2 1 2 2 2 2 1 1 2 2 2 2 2 2 1 1 1 2 2 2 1
## [112] 1 2 2 1 1 1 1 1 2 2 2 2 2 2 2 1 2 2 2 1 1 1 2 1 2 2 2 1 2 2 2 2 1 2 2 1 2
## [149] 2 2 2 2 2 2 2 2 2 2 2 2 1 2 1 2 2 1 2 2 1 2 2 2 2 2 2 1 2 2 2 2 2 1 2 2
## [186] 2 1 2 1 1 1 2 2 2 2 2 1 2 2 1 2 1 1 2 1 2 2 2 2 2 2 2 2 1 1 1 2 2 2 1 2 2
## [223] 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [260] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2
## [297] 1 1 2 1 1 1 1 1 1 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [334] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [371] 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 1 2 2 1 2 2 1 1 1 2 2 2 2 2 2 2 2 2 2 2
## [408] 1 2 2 1 1 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2
## [445] 2 1 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2
## [482] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 1
## Objective function:
##    build     swap
## 4.125728 4.125728
##
## Available components:
##  [1] "medoids"    "id.med"     "clustering" "objective"  "isolation"
##  [6] "clusinfo"   "silinfo"    "diss"       "call"       "data"
```

```r
fviz_nbclust(clusterdat, pam, method = "wss")
```



```
# look at elbow --> three clusters

# another approach
gap_stat <- clusGap(clusterdat, FUN = pam, K.max = 15,B = 50)
fviz_gap_stat(gap_stat)
```

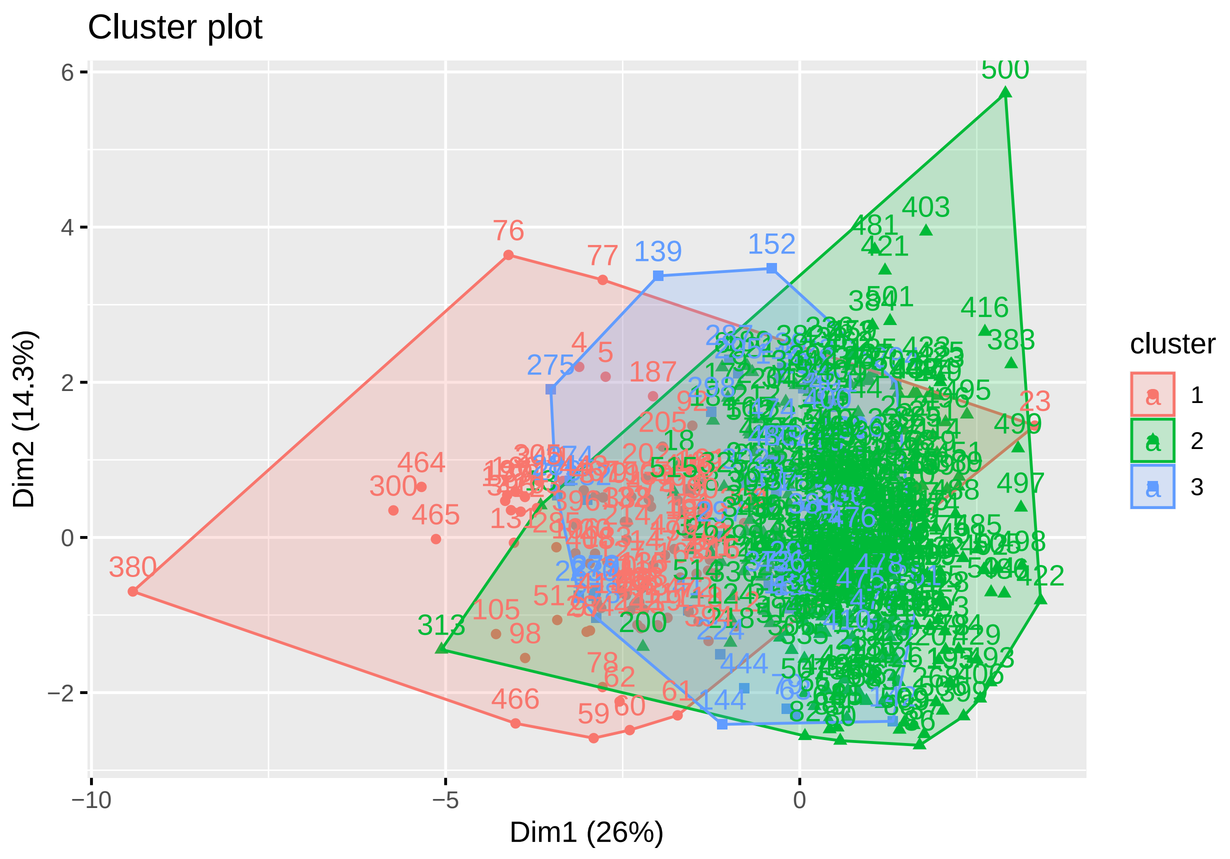


```r
# choosing 3 clusters seem fine?
set.seed(1)
kmed <- pam(clusterdat, k = 3)
kmed
```

```
## Medoids:
##       ID X Y FFMC   DMC    DC  ISI temp RH wind rain log_area
## [1,]  40 4 4 88.1  25.7  67.6  3.8 14.1 43  2.7    0 0.000000
## [2,]  14 6 5 90.9 126.5 686.5  7.0 21.3 42  2.2    0 0.000000
## [3,] 478 4 3 93.7 101.3 423.4 14.7 26.1 45  4.0    0 2.123458
## Clustering vector:
##   [1] 1 2 2 1 1 3 3 2 2 2 2 2 2 2 2 2 1 2 1 1 2 2 1 3 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [38] 2 2 1 3 2 2 2 2 2 2 3 1 1 2 2 2 2 2 2 2 2 1 1 1 1 3 2 2 2 2 2 2 1 1 2 1 2
##  [75] 2 1 1 1 3 2 2 2 2 2 2 2 2 2 2 1 2 1 3 2 2 2 1 1 2 2 2 2 2 2 1 1 1 2 2 2 1
## [112] 1 2 2 1 1 1 1 1 3 2 2 2 2 2 2 1 2 2 2 1 1 1 2 1 2 2 2 3 2 2 3 3 3 3 2 1 2
## [149] 2 2 3 3 3 2 2 2 2 2 2 2 1 2 1 2 2 1 2 2 1 2 2 2 2 2 2 1 2 2 2 2 2 1 2 2
## [186] 2 1 2 1 1 1 2 2 2 2 2 1 2 2 2 2 1 1 2 1 2 2 2 2 2 2 2 2 1 1 1 2 2 2 1 2 2
## [223] 1 3 2 2 2 3 2 3 2 2 2 2 2 2 2 2 2 1 1 1 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2
## [260] 2 3 2 2 2 2 3 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 1 1 3 3 3 3 3 3 3 3 3 3 3
## [297] 3 3 3 1 3 3 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [334] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [371] 2 3 2 2 2 2 2 2 1 1 3 2 2 2 2 2 2 1 2 2 1 2 2 1 1 1 2 2 2 3 3 2 2 2 2 2 2
## [408] 1 2 3 1 1 3 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 1 3
## [445] 2 1 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 3 3 3 3 3 3 3 2
## [482] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1
## Objective function:
```

```
##    build     swap
## 77.72867 71.36682
##
## Available components:
##  [1] "medoids"    "id.med"     "clustering" "objective"  "isolation"
##  [6] "clusinfo"   "silinfo"    "diss"       "call"       "data"
```

```
fviz_cluster(kmed, data = clusterdat)
```



```
# combine clusters into the dataset
labeled_data = cbind(clusterdat, cluster = kmed$cluster)
head(labeled_data)
```

```
##   X Y FFMC  DMC    DC  ISI temp RH wind rain log_area cluster
## 1 7 5 86.2 26.2  94.3  5.1  8.2 51  6.7  0.0        0       1
## 2 7 4 90.6 35.4 669.1  6.7 18.0 33  0.9  0.0        0       2
## 3 7 4 90.6 43.7 686.9  6.7 14.6 33  1.3  0.0        0       2
## 4 8 6 91.7 33.3  77.5  9.0  8.3 97  4.0  0.2        0       1
## 5 8 6 89.3 51.3 102.2  9.6 11.4 99  1.8  0.0        0       1
## 6 8 6 92.3 85.3 488.0 14.7 22.2 29  5.4  0.0        0       3
```

```
summary(lm(log_area ~ as.factor(cluster), data = labeled_data))
```

```
##
```

```
## Call:
## lm(formula = log_area ~ as.factor(cluster), data = labeled_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.2338 -1.1595 -0.7576  0.8985  5.8362
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)           0.8391     0.1447   5.800 1.16e-08 ***
## as.factor(cluster)2   0.3204     0.1623   1.974   0.0489 *
## as.factor(cluster)3   0.3947     0.2266   1.742   0.0821 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.395 on 514 degrees of freedom
## Multiple R-squared:  0.00861,    Adjusted R-squared:  0.004752
## F-statistic: 2.232 on 2 and 514 DF,  p-value: 0.1084
```

```
# significant....

# Look into these clusters
mod_cl1 <-lm(log_area ~ ., data = labeled_data %>%
               filter(cluster == 1) %>%
               select(-c(12)))
summary(mod_cl1)
```

```
##
## Call:
## lm(formula = log_area ~ ., data = labeled_data %>% filter(cluster ==
##     1) %>% select(-c(12)))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.4213 -0.7640 -0.4369  0.5477  3.7269
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.371747   2.408424   0.985  0.32763
## X            0.266030   0.084108   3.163  0.00219 **
## Y           -0.280369   0.116925  -2.398  0.01876 *
## FFMC        -0.005621   0.024591  -0.229  0.81977
## DMC          0.022408   0.018955   1.182  0.24056
## DC          -0.009456   0.008204  -1.153  0.25243
## ISI         -0.022330   0.025365  -0.880  0.38125
## temp        -0.054115   0.048050  -1.126  0.26335
## RH          -0.006104   0.010261  -0.595  0.55357
## wind         0.012497   0.071996   0.174  0.86262
## rain        -5.617550   6.374745  -0.881  0.38077
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.186 on 82 degrees of freedom
## Multiple R-squared:  0.1788, Adjusted R-squared:  0.0787
## F-statistic: 1.786 on 10 and 82 DF,  p-value: 0.07608
```

```r
mod_cl2 <-lm(log_area ~ ., data = labeled_data %>%
               filter(cluster == 2) %>%
               select(-c(12)))
summary(mod_cl2)
```

```
##
## Call:
## lm(formula = log_area ~ ., data = labeled_data %>% filter(cluster ==
##     2) %>% select(-c(12)))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9154 -1.0856 -0.5995  0.8381  5.5632
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.8968932  2.7356368  -0.328   0.7432
## X           -0.0179912  0.0384893  -0.467   0.6405
## Y            0.1323925  0.0774968   1.708   0.0885 .
## FFMC         0.0075077  0.0262507   0.286   0.7750
## DMC          0.0007332  0.0017004   0.431   0.6666
## DC           0.0007052  0.0013523   0.522   0.6023
## ISI         -0.0190621  0.0256045  -0.744   0.4571
## temp         0.0223441  0.0268539   0.832   0.4059
## RH          -0.0048081  0.0083604  -0.575   0.5656
## wind         0.0623466  0.0516119   1.208   0.2279
## rain         0.0798122  0.2227726   0.358   0.7204
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.427 on 349 degrees of freedom
## Multiple R-squared:  0.0262, Adjusted R-squared:  -0.001702
## F-statistic: 0.939 on 10 and 349 DF,  p-value: 0.4972
```

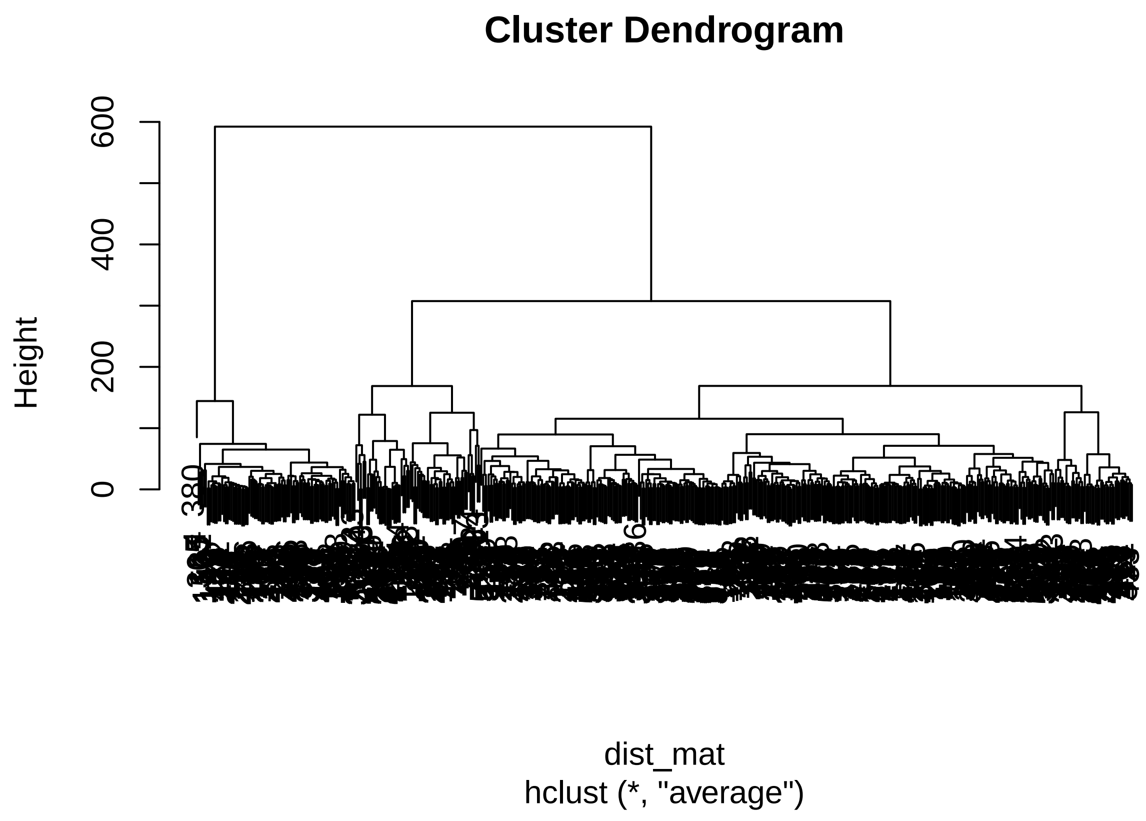
```r
mod_cl3 <-lm(log_area ~ ., data = labeled_data %>%
               filter(cluster == 3) %>%
               select(-c(12)))
summary(mod_cl3)
```

```
##
## Call:
## lm(formula = log_area ~ ., data = labeled_data %>% filter(cluster ==
##     3) %>% select(-c(12)))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.8816 -0.9452 -0.0847  0.5522  3.6009
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -2.684879  10.130172  -0.265   0.7920
## X             0.152305   0.087245   1.746   0.0867 .
## Y            -0.135775   0.154762  -0.877   0.3843
## FFMC          0.044470   0.116231   0.383   0.7035
## DMC           0.002317   0.006565   0.353   0.7256
## DC           -0.002733   0.002619  -1.044   0.3014
## ISI          -0.100227   0.095833  -1.046   0.3004
## temp          0.006374   0.048834   0.131   0.8966
## RH            0.009315   0.015453   0.603   0.5492
## wind          0.231144   0.119205   1.939   0.0578 .
## rain        -11.363667   8.476510  -1.341   0.1858
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.41 on 53 degrees of freedom
## Multiple R-squared:  0.1869, Adjusted R-squared:  0.03346
## F-statistic: 1.218 on 10 and 53 DF,  p-value: 0.3012
```

```r
# Not as informative
```

Hierarchical clustering

```r
dist_mat <- dist(newdata, method = 'euclidean')
hclust_avg <- hclust(dist_mat, method = 'average')
plot(hclust_avg)
```



```r
cut <- cutree(hclust_avg, k = 3)
labeled_data = cbind(newdata, cluster = cut)
summary(lm(log_area ~ as.factor(cluster), data = labeled_data))
```

```
##
## Call:
## lm(formula = log_area ~ as.factor(cluster), data = labeled_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.1600 -1.1595 -0.7407  0.9150  5.8362
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)            0.8745     0.1489   5.872 7.73e-09 ***
## as.factor(cluster)2   0.2850     0.1661   1.715   0.0869 .
## as.factor(cluster)3   0.2855     0.2246   1.271   0.2043
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.397 on 514 degrees of freedom
## Multiple R-squared:  0.005881,   Adjusted R-squared:  0.002012
## F-statistic:  1.52 on 2 and 514 DF,  p-value: 0.2196
```