Global Poweplants Fuel Analysis

Ridhiwan Mseya

02/12/2022

1. INTRODUCTION

The world is in need of reliable and affordable green power sector. The shift towards cleaner energy that is environmentally sustainable forces us to rethink and reinvent better sources of energy to power our societies. To do this we first have to take a zoomed out look at the sources of energy around the world, the level of dependency towards those sources and their effects on the environment. Governments and companies will now make decisions that will depend on the power sector and its transformation towards green energy. For example, if a carbon tax is to be passed to certain sources of energy then the price of electricity will increase and affect prices of products and thus the economy.

In this analysis we will use "The Global Plant Database" which is an open source and open access dataset of grid scale(1MW and greater) electricity generating facilities in the world. This dataset consists of the country, name of the power plant, the capacity in megawatts, the latitude and longitude of the plant, the primary fuel for power generation, year of commission, the generated electricity in gigawatt hours and more. We will incorporate the available data into a predictive model that guesses the primary fuel of prospective plant anywhere in the world based on its location, capacity and other parameters available in the data. Below is a quick view of our database:

```r
library(tibble)
library(readr)
library(tidyverse)

gppb <- read.csv("global_power_plant_database.csv", header = TRUE)

gppb1 <- tibble(gppb)

head(gppb1)
```

```
## # A tibble: 6 x 24
##   country country_long name             gppd_idnr capacity_mw latitude longitude
##   <chr>   <chr>        <chr>            <chr>           <dbl>    <dbl>     <dbl>
## 1 AFG     Afghanistan  Kajaki Hydroele~ GEODB004~        33       32.3      65.1
## 2 AFG     Afghanistan  Mahipar Hydroel~ GEODB004~        66       34.6      69.5
## 3 AFG     Afghanistan  Naghlu Dam Hydr~ GEODB004~       100       34.6      69.7
## 4 AFG     Afghanistan  Nangarhar (Daru~ GEODB004~        11.6     34.5      70.4
## 5 AFG     Afghanistan  Northwest Kabul~ GEODB004~        42       34.6      69.1
## 6 AFG     Afghanistan  Pul-e-Khumri Hy~ GEODB004~         6       35.9      68.7
## # ... with 17 more variables: primary_fuel <chr>, other_fuel1 <chr>,
## #   other_fuel2 <chr>, other_fuel3 <chr>, commissioning_year <dbl>,
## #   owner <chr>, source <chr>, url <chr>, geolocation_source <chr>,
## #   wepp_id <chr>, year_of_capacity_data <int>, generation_gwh_2013 <dbl>,
## #   generation_gwh_2014 <dbl>, generation_gwh_2015 <dbl>,
## #   generation_gwh_2016 <dbl>, generation_gwh_2017 <dbl>,
## #   estimated_generation_gwh <dbl>
```

The ability to predict the type of fuel that could be used in a given location is essential in planning the type of emissions and waste that could be associated with electricity generation in that area. A pre-plan on how to mitigate the effects of the emissions and waste products will help in making the production process greener. New environment friendly technology can be developed years ahead of electricity production and specific to that location.

2. METHODS

In this section we will visualize some aspects of our data, do some correlations and clustering tests, clean the data, alter its form if necessary and make a model using different methods like kmeans and random forest.

Clean the Data

Before we perform any analysis we will first remove all the data that may cause inconsistency in our analysis.Some of the information in the columns in our data are not necessary for our analysis i.e. name of the country, name of power plant, other fuels, year of data generation.

```
# Choose columns wit the most essential data

gppb2 <- gppb1 %>% select(gppd_idnr,capacity_mw,latitude,longitude,primary_fuel,
                          generation_gwh_2013,generation_gwh_2014,generation_gwh_2015,
                          generation_gwh_2016,generation_gwh_2017,estimated_generation_gwh)

head(gppb2)
```

```
## # A tibble: 6 x 11
##   gppd_idnr    capacity_mw latitude longitude primary_fuel generation_gwh_2013
##   <chr>              <dbl>    <dbl>     <dbl> <chr>                      <dbl>
## 1 GEODB0040538        33       32.3      65.1 Hydro                         NA
## 2 GEODB0040541        66       34.6      69.5 Hydro                         NA
## 3 GEODB0040534       100       34.6      69.7 Hydro                         NA
## 4 GEODB0040536        11.6     34.5      70.4 Hydro                         NA
## 5 GEODB0040540        42       34.6      69.1 Gas                           NA
## 6 GEODB0040537         6       35.9      68.7 Hydro                         NA
## # ... with 5 more variables: generation_gwh_2014 <dbl>,
## #   generation_gwh_2015 <dbl>, generation_gwh_2016 <dbl>,
## #   generation_gwh_2017 <dbl>, estimated_generation_gwh <dbl>
```

```
# Assume the energy production was zero in the years that it was not recorded
gppb2[is.na(gppb2)] <- 0
head(gppb2)
```

```
## # A tibble: 6 x 11
##   gppd_idnr    capacity_mw latitude longitude primary_fuel generation_gwh_2013
##   <chr>              <dbl>    <dbl>     <dbl> <chr>                      <dbl>
## 1 GEODB0040538        33       32.3      65.1 Hydro                          0
## 2 GEODB0040541        66       34.6      69.5 Hydro                          0
## 3 GEODB0040534       100       34.6      69.7 Hydro                          0
## 4 GEODB0040536        11.6     34.5      70.4 Hydro                          0
## 5 GEODB0040540        42       34.6      69.1 Gas                            0
## 6 GEODB0040537         6       35.9      68.7 Hydro                          0
## # ... with 5 more variables: generation_gwh_2014 <dbl>,
## #   generation_gwh_2015 <dbl>, generation_gwh_2016 <dbl>,
## #   generation_gwh_2017 <dbl>, estimated_generation_gwh <dbl>
```

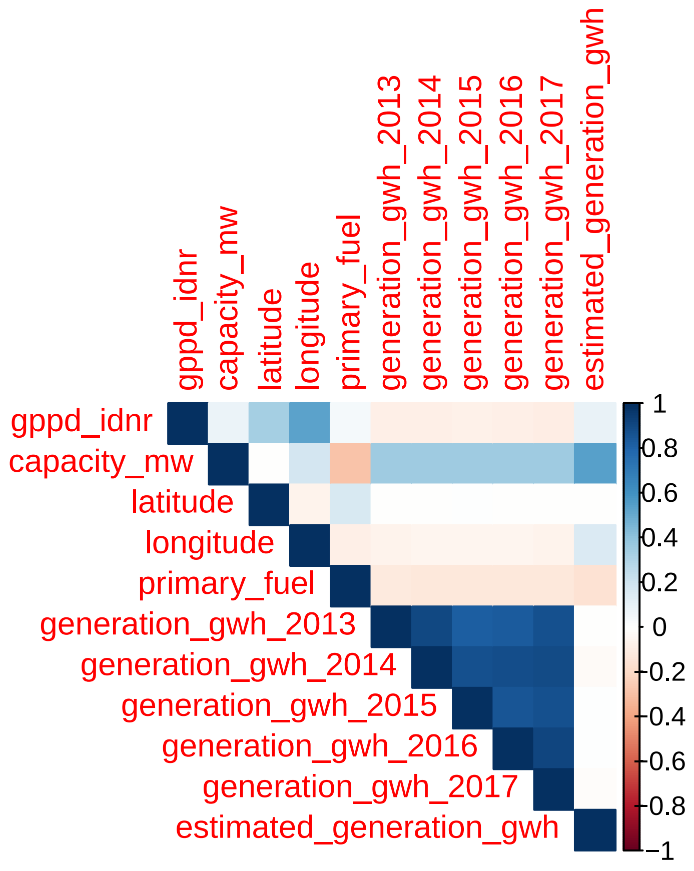
Correlation

Correlation analysis of the variables in our database.

```
library(corrplot)

# factorize the character columns
gppb3 <- gppb2
gppb3$gppd_idnr <- as.numeric(as.factor(gppb3$gppd_idnr))
gppb3$primary_fuel <- as.numeric(as.factor(gppb3$primary_fuel))
head(gppb3)
```

```
## # A tibble: 6 x 11
##   gppd_idnr capacity_mw latitude longitude primary_fuel generation_gwh_2013
##       <dbl>       <dbl>    <dbl>     <dbl>        <dbl>               <dbl>
## 1      7014          33     32.3      65.1            6                   0
## 2      7016          66     34.6      69.5            6                   0
## 3      7010         100     34.6      69.7            6                   0
## 4      7012          11.6   34.5      70.4            6                   0
## 5      7015          42     34.6      69.1            4                   0
## 6      7013           6     35.9      68.7            6                   0
## # ... with 5 more variables: generation_gwh_2014 <dbl>,
## #   generation_gwh_2015 <dbl>, generation_gwh_2016 <dbl>,
## #   generation_gwh_2017 <dbl>, estimated_generation_gwh <dbl>
```

```
# Plot the correlation figure
corrplot(cor(gppb3),method = "color",type = "upper")
```



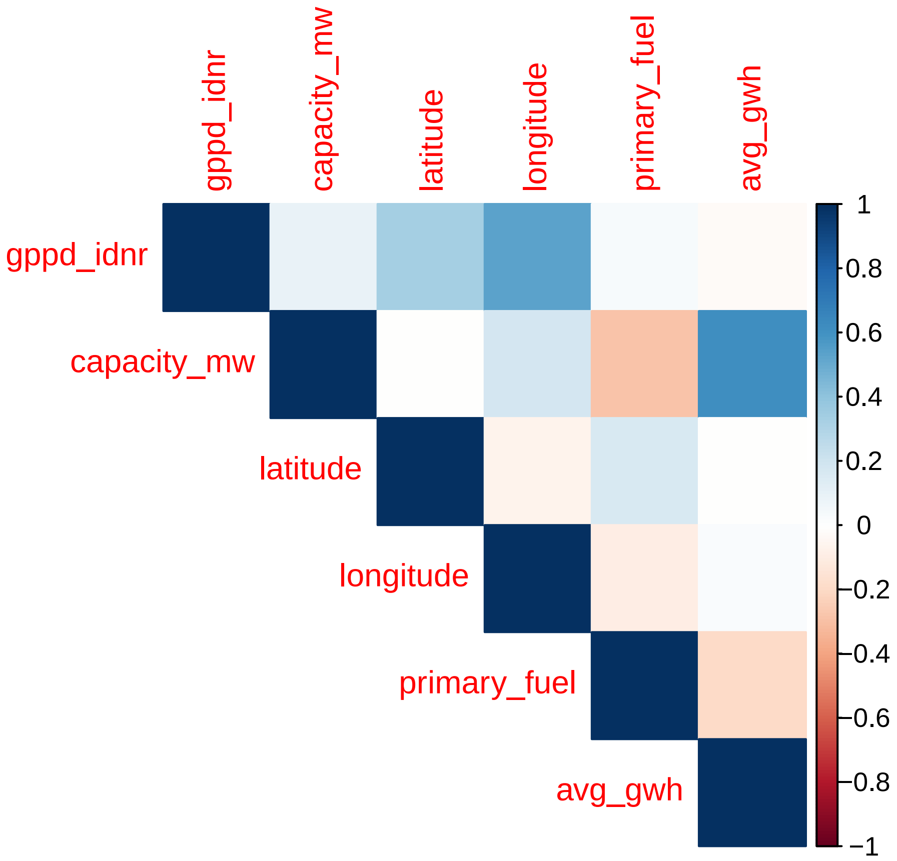
There is very little correlation between primary fuel and the latitude and longitudes of the powerplant. This is good because our model based on these parameters will have less bias. We also observe very high correlation between the generated energy from 2013 to 2017 because the capacity of the powerplant increases slightly in the scale of gigawatts throughout the years. We will remake these 5 energy columns into one column by finding the average over the 5 years.

```
# Average the energy generation columns into one

gppb3$avg_gwh <- rowMeans(gppb3[,c(6:11)], na.rm = TRUE)

gppb4 <- gppb3 %>% select(-c(generation_gwh_2013,generation_gwh_2014,
                             generation_gwh_2015,generation_gwh_2016,generation_gwh_2017,
                             estimated_generation_gwh))
head(gppb4)
```

```
## # A tibble: 6 x 6
##   gppd_idnr capacity_mw latitude longitude primary_fuel avg_gwh
##       <dbl>       <dbl>    <dbl>     <dbl>        <dbl>   <dbl>
## 1      7014        33       32.3      65.1            6       0
## 2      7016        66       34.6      69.5            6       0
## 3      7010       100       34.6      69.7            6       0
## 4      7012        11.6     34.5      70.4            6       0
## 5      7015        42       34.6      69.1            4       0
## 6      7013         6       35.9      68.7            6       0
```

At this point any power plant that displays an average energy generation of zero should be removed because it has missing data for all the given years and will affect our model negatively. Then we will re-plot our correlation plot.

```
# remove rows with zero average energy generation
gppb5 <- gppb4[gppb4$avg_gwh != 0,]

#plot new correlation plot
corrplot(cor(gppb5),method = "color",type = "upper")
```

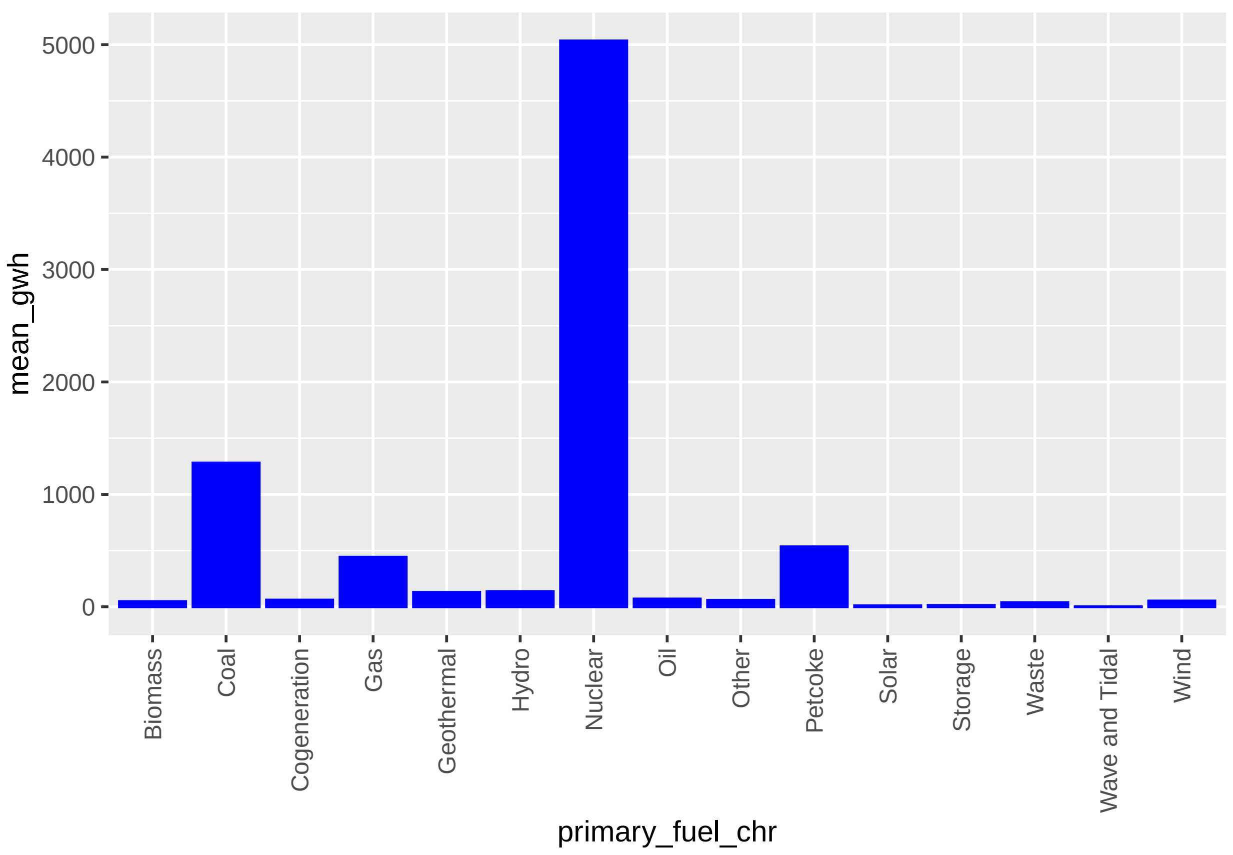


In this plot there is stronger correlation between average energy generation and the capacity of the power plant which makes sense.There is also an interesting negative correlation between the capacity of the power plant and the primary fuel. Same goes for primary fuel and the average energy generated. These could mean that our generation power and capacity strongly depend on the primary fuel, which seems obvious.

Visualization

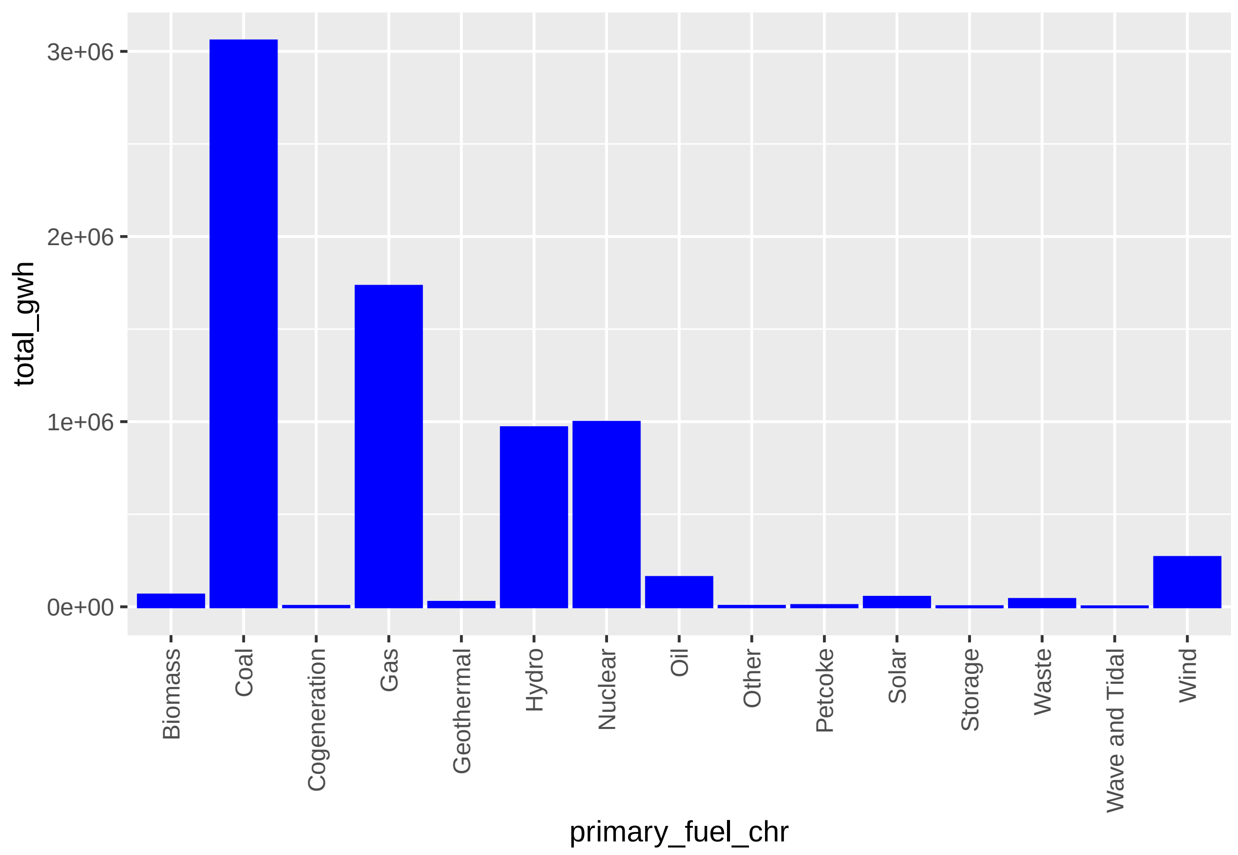
We will investigate some of the features observable in our data. Let us start with the mean energy that each primary fuel has produced all over the world.

```
library(ggplot2)

# Add a column with string characters for primary fuel
gppb4$primary_fuel_chr <- add_column(gppb2$primary_fuel)

# Create a plot
gppb4 %>% select(avg_gwh,primary_fuel_chr) %>% group_by(primary_fuel_chr) %>%
  summarise(mean_gwh = mean(avg_gwh)) %>%
  ggplot(aes(primary_fuel_chr,mean_gwh)) +
  geom_bar(stat = "identity", color = "blue", fill = "blue") +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))
```



We can observe that on average the most power producing plants are those with the primary fuel of nuclear, Coal, petcoke and gas. Nuclear energy has very high output per powerplant compared to any of the other fuels. Now let us view the most total energy produced by each fuel around the world.

```
# Create a plot
gppb4 %>% select(avg_gwh,primary_fuel_chr) %>% group_by(primary_fuel_chr) %>%
  summarise(total_gwh = sum(avg_gwh)) %>%
  ggplot(aes(primary_fuel_chr,total_gwh)) +
  geom_bar(stat = "identity", color = "blue", fill = "blue") +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))
```



In terms of the total energy produced Coal leads the way followed by Gas. Hydro and Nuclear almost have the same contribution. The rest of the renewable energy sources have little contribution into powering the world.

In order to do this we will first need to do some stratification of our data.The strata of the rounded off longitude and latitudes will cover a greater region than the specific position provided by the original database.

```r
# Round off the latitude and longitudes into integers for stratification
gppb6 <- gppb5
gppb6$longitude <- round(gppb6$longitude)
gppb6$latitude <- round(gppb6$latitude)
head(gppb6)
```

```
## # A tibble: 6 x 6
##   gppd_idnr capacity_mw latitude longitude primary_fuel avg_gwh
##       <dbl>       <dbl>    <dbl>     <dbl>        <dbl>   <dbl>
## 1     17741          27       40        20            6   14.9
## 2     17742         500       42        20            6  275.
## 3     17743         600       42        20            6  330.
## 4     17744           5       41        20            6    2.75
## 5     17745          24       42        20            6   13.2
## 6     17746          25       42        20            6   13.8
```

K-Means Clustering

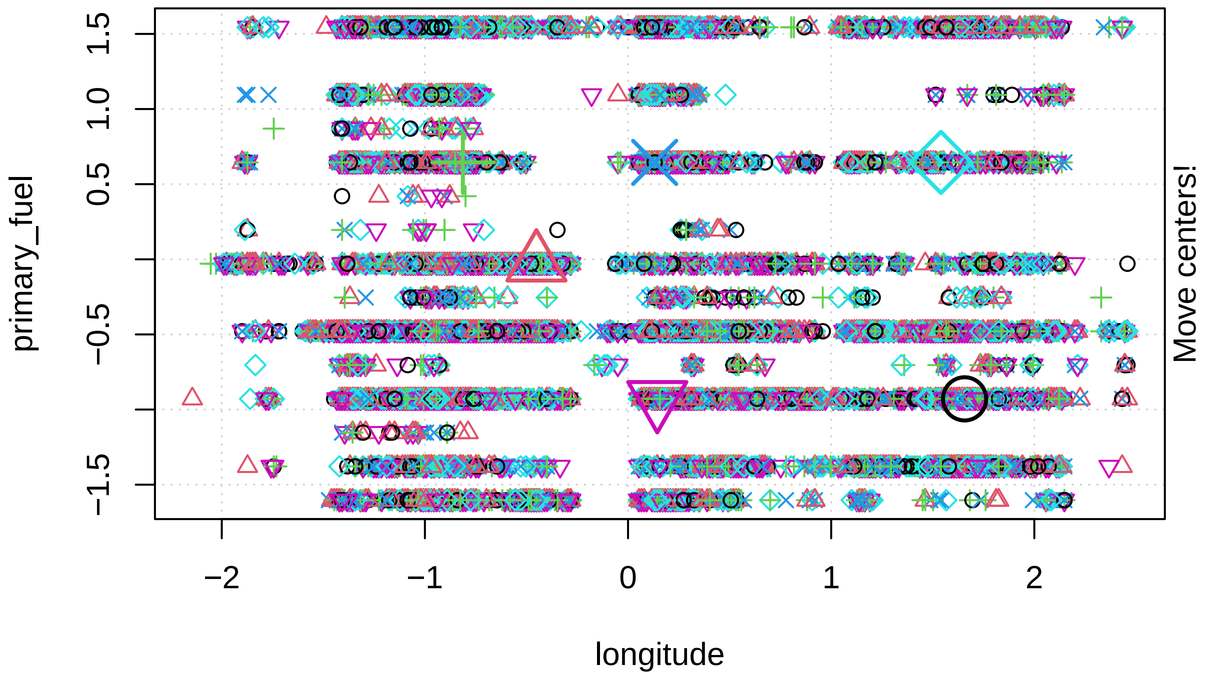
The first model will be unsupervised in which we will do clustering using the k-means algorithm. We will get rid of the id,longitude and latitude columns of the powerplants and rescale all the values of the remaining columns.

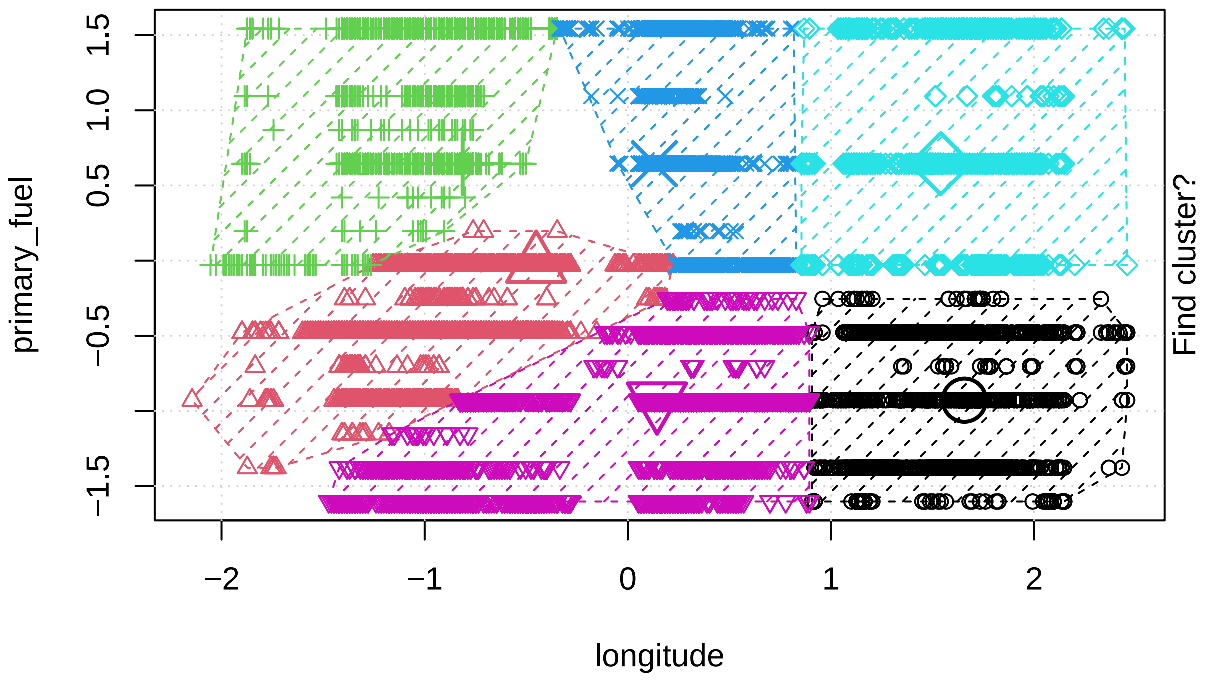
```
#edit and rescale the dataset
gppb7 <- gppb6 %>% mutate(capacity_mw = scale(capacity_mw),
                          longitude = scale(longitude),
                          primary_fuel = scale(primary_fuel),
                          avg_gwh = scale(avg_gwh)) %>% select(-c(gppd_idnr,latitude))

head(gppb7)
```

```
## # A tibble: 6 x 4
##   capacity_mw[,1] longitude[,1] primary_fuel[,1] avg_gwh[,1]
##             <dbl>         <dbl>            <dbl>       <dbl>
## 1          -0.307         0.428           -0.479     -0.215
## 2           0.591         0.428           -0.479      0.0192
## 3           0.781         0.428           -0.479      0.0686
## 4          -0.349         0.428           -0.479     -0.225
## 5          -0.313         0.428           -0.479     -0.216
## 6          -0.311         0.428           -0.479     -0.215
```

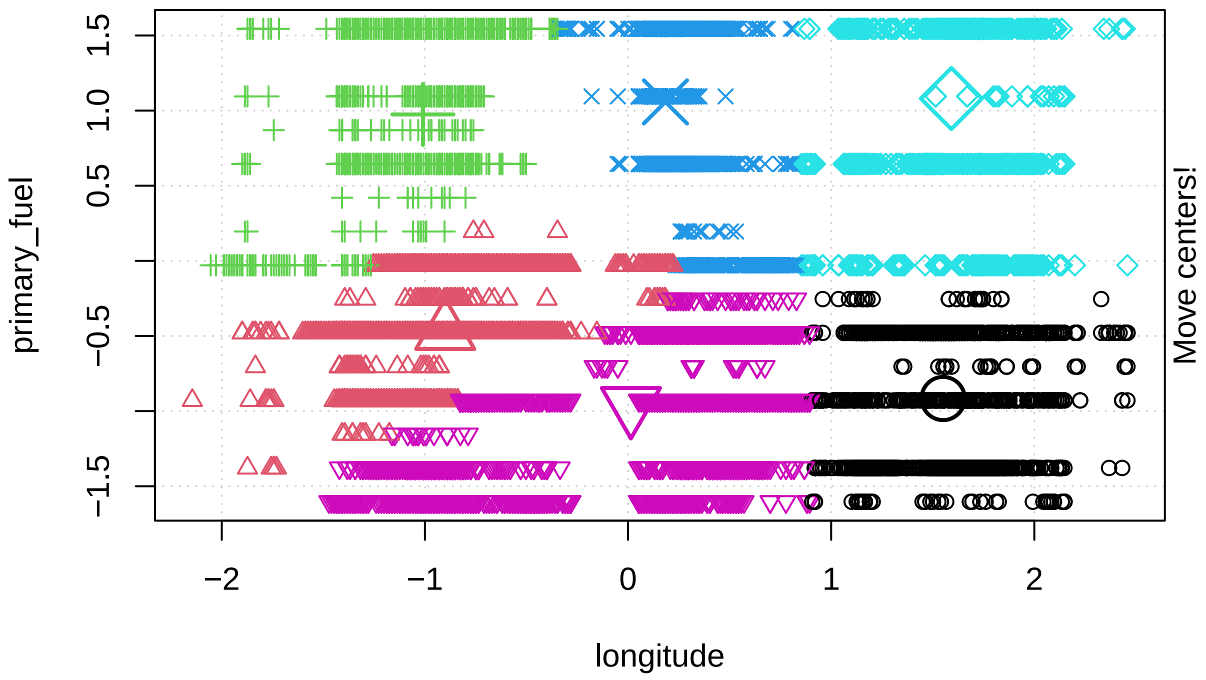
Let us animate the clustering using longitude and primary fuel to observe the process of clustering and the type of separation that is possible. We will choose the number of clusters to be less than half the number of different fuel types for illustration purposes.

```
# Make the first animation of longitude and primary fuel
set.seed(2233)
library(animation)
ani.options(nmax = 3)
kmeans.ani(gppb7[2:3],6, pch = 1:6, col = 1:6)
```

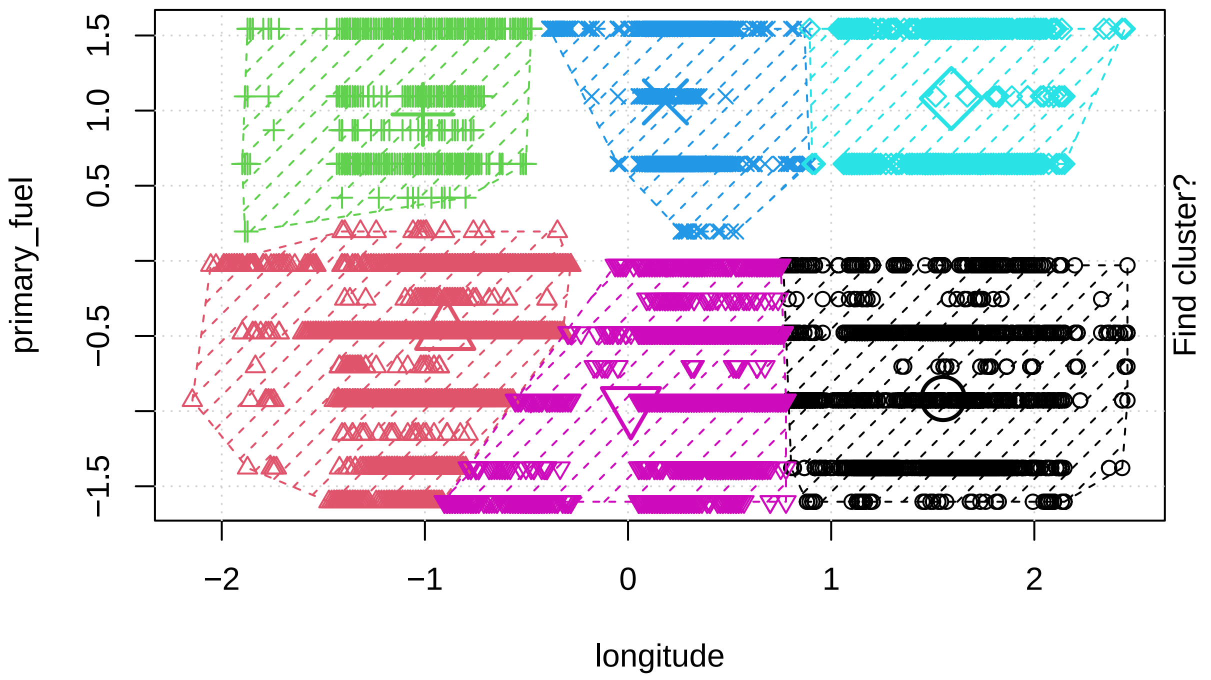


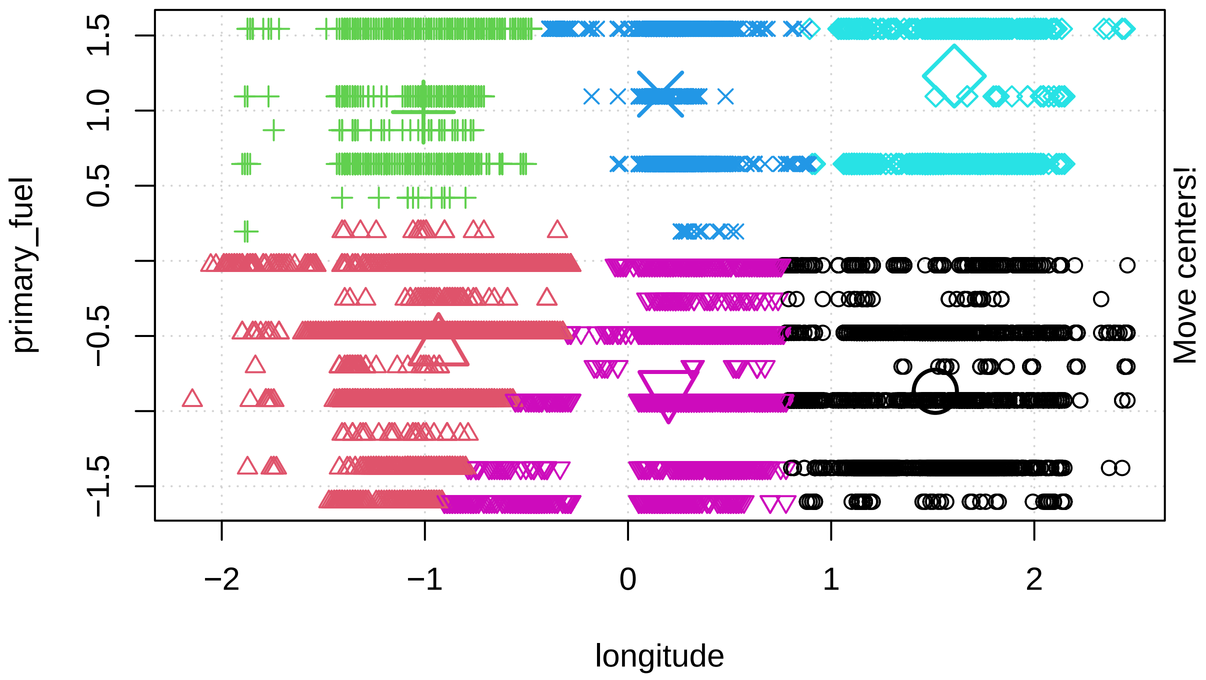


primary_fuel
1.5
1.0
0.5
−0.5
−1.5
−2
−1
0
1
2
longitude
Find cluster?

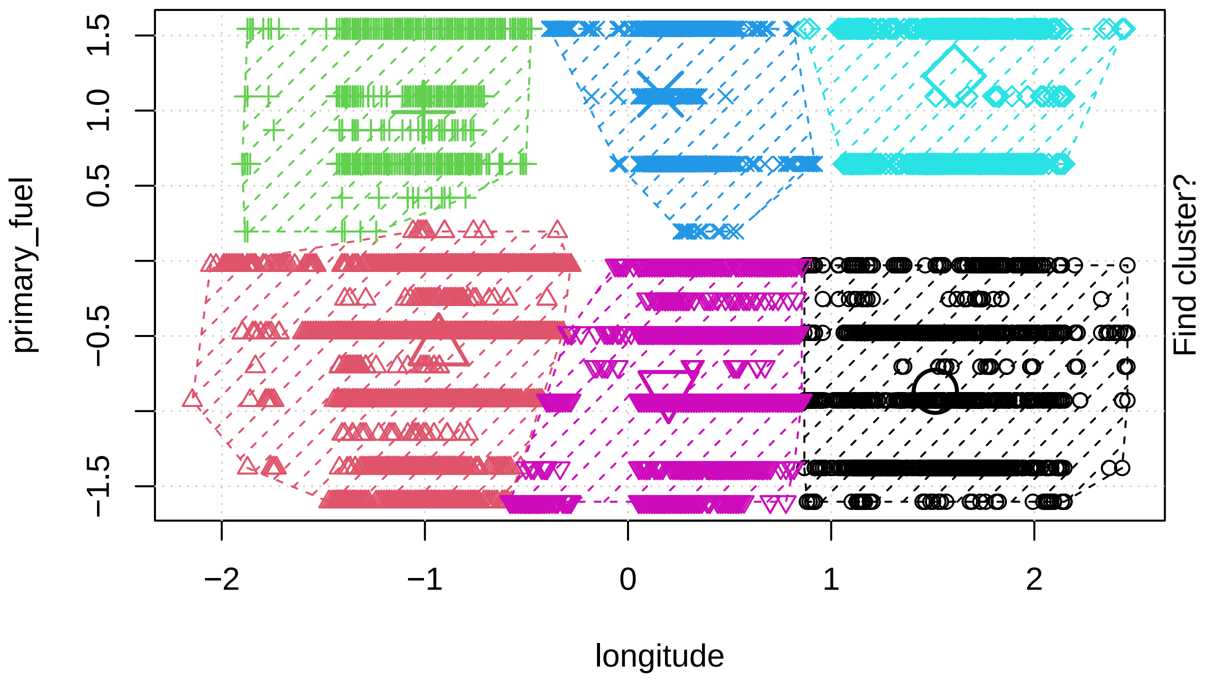


primary_fuel
1.5
1.0
0.5
−0.5
−1.5
−2
−1
0
1
2
longitude
Move centers!


primary_fuel
longitude
Find cluster?
1.5
1.0
0.5
−0.5
−1.5
−2
−1
0
1
2



primary_fuel
1.5
1.0
0.5
−0.5
−1.5
−2
−1
0
1
2
longitude
Move centers!

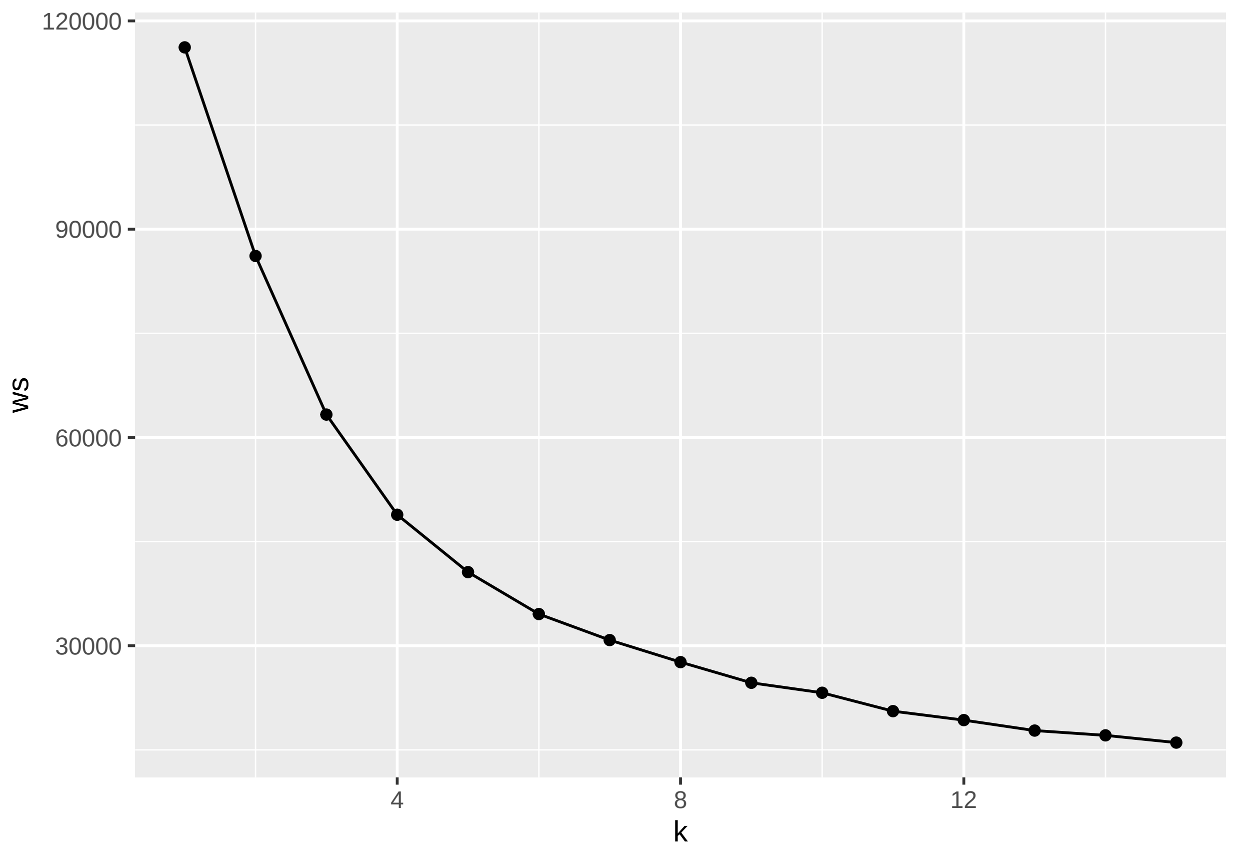


As it has been shown above from the many steps that the k-means algorithm takes to move centers and find clusters, our data can be clustered but knowing the number of clusters to start with is not easy. We will use a strategy that will give us a graph that will help us choose the most optimal number of clusters to use for our analysis. The elbow method is a good method for large datasets as it looks at the variance as a function of the number of clusters.

```
# Analyse the best number of clusters
k.max <- 15
data <- gppb7
wss <- sapply(1:k.max,
              function(k){kmeans(data, k, nstart=50,iter.max = 15 )$tot.withinss})
wss
```

```
##  [1] 116184.00  86139.47  63292.73  48867.42  40599.54  34560.34  30811.79
##  [8]  27637.14  24659.99  23221.79  20574.87  19284.14  17778.51  17088.26
## [15]  16045.58
```

```
# plot total within sum of squares for each cluster count
tab <- cbind.data.frame(1:k.max,wss)
colnames(tab) <- c("k","ws")
tab %>% ggplot(aes(k, ws)) + geom_line() +geom_point()
```



6 looks like the optimal number of clusters for our data as the change seems to be minimal as we keep adding more clusters after that point.

```
# Redo kmeans with optimal k
set.seed(2233)
clst <- kmeans(gppb7,6)
clst$size
```

```
## [1]  1793 10030  4377  4161  7990   696
```

```
centr <- clst$centers
centr
```

```
##   capacity_mw  longitude primary_fuel     avg_gwh
## 1  -0.2844026  1.5957723    1.1664685 -0.19919432
## 2  -0.3121407 -0.3749132    1.0580820 -0.20064969
## 3  -0.1809708  0.4059754   -0.7357860 -0.16329171
## 4   0.5731941  1.4558923   -0.8829858  0.14628437
## 5  -0.1319967 -0.8970465   -0.6428950 -0.07505392
## 6   4.4574803  0.3328144   -0.9664807  4.41866544
```

From the cluster size we can see that there is some sort of homogeinity in the clusters which might be good. Let us dig deeper into the centers and get a peek of how the features stack up into the clusters.

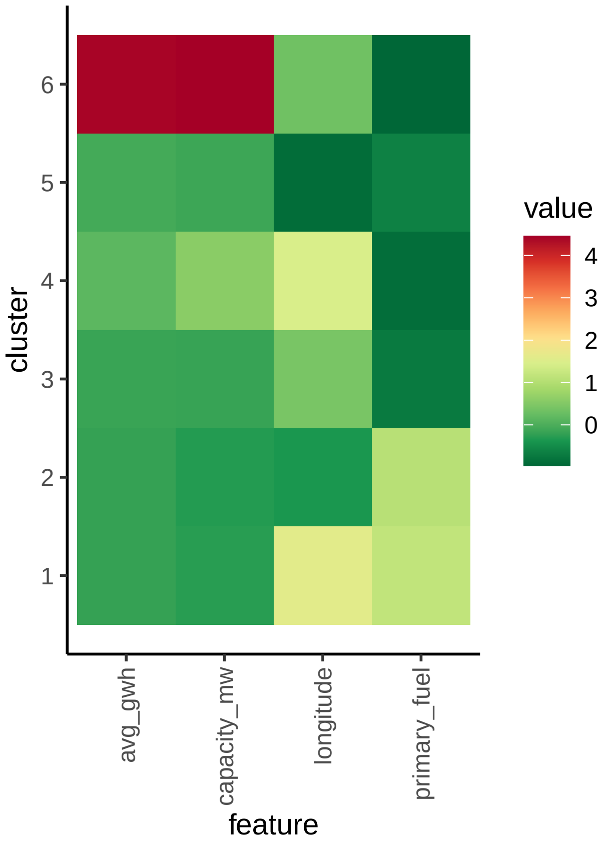
```
# create dataset for reshape
cluster <- c(1: 6)
centered <- data.frame(cluster, centr)

# Reshape the data
```

```r
reshaped <- gather(centered, feature, value, capacity_mw: avg_gwh)
head(reshaped,10)
```

```
##    cluster     feature      value
## 1        1 capacity_mw -0.2844026
## 2        2 capacity_mw -0.3121407
## 3        3 capacity_mw -0.1809708
## 4        4 capacity_mw  0.5731941
## 5        5 capacity_mw -0.1319967
## 6        6 capacity_mw  4.4574803
## 7        1   longitude  1.5957723
## 8        2   longitude -0.3749132
## 9        3   longitude  0.4059754
## 10       4   longitude  1.4558923
```

Let us now create the heatmap to visualize the intensity of features in clusters.

```r
library(RColorBrewer)
# Create the palette
heat.palette <-colorRampPalette(rev(brewer.pal(10, 'RdYlGn')),space='Lab')

#make the plot of the heatmap
reshaped %>% ggplot(aes(x = feature, y = cluster, fill = value)) +
    scale_y_continuous(breaks = seq(1, 6, by = 1)) +
    geom_tile() +
    coord_equal() +
    scale_fill_gradientn(colours = heat.palette(90)) +
    theme_classic() +
    theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))
```



The sixth cluster which was the smallest cluster by size, has a high intensity of two features that are related to energy production while the rest of the clusters have an almost homogeneous distribution of the features in each cluster. The latitudes with unsupervised learning can be grouped into 6 clusters.

Introduction to Models

The models that we will develop have a goal of predicting the most probable primary fuel to be used at a given location once we know the latitude and longitude and other parameters specific to the powerplant.

Random Forest

Random forest is a supervised learning approach that uses decisions of the many to predict the most probable result for a given model. For our dataset we will train a model and choose the best parameters for prediction accordingly.

Let us start by training our model on small different datasets and then combine their results. This will need us to split our data into test and train sets since I have a lot of observables around 29,000 thus I can split my data by 80/20 or 90/10 or somewhere in between as the variance of my parameters will be okay with either of the two. For the purpose of easier computation, I will use the 80/20 split.

```
library(caret)
library(e1071)

# Split the data into train and test sets
# Here we use the version of our datasets that contains-
# all important columns with original values
set.seed(2233)
test_ind <- createDataPartition(as.factor(gppb5$primary_fuel),p=0.2
```

```r
                                ,list = FALSE, times = 1)

test_data <- gppb5[test_ind,]
train_data <- gppb5[-test_ind,]

head(test_data)
```

```
## # A tibble: 6 x 6
##   gppd_idnr capacity_mw latitude longitude primary_fuel avg_gwh
##       <dbl>       <dbl>    <dbl>     <dbl>        <dbl>   <dbl>
## 1     17743         600     42.1     19.8             6   330.
## 2     22110          71     36.9      7.76            4    49.0
## 3     22101         292     35.8      7.36            4   201.
## 4     22106         200     31.8      6.05            4   138.
## 5     22091         588     36.8      5.88            4   406.
## 6     22115         396     34.5      6.98            4   273.
```

```r
library(randomForest)
library(doParallel)

set.seed(1558)

# set up parallelism
cores <- 3
clt <- makePSOCKcluster(cores)
registerDoParallel(clt)

# choose parameters for training
trControl <- trainControl(method = "cv",
    number = 7,
    search = "grid")

# tune the grid
tunegrid = expand.grid(.mtry=c(1:20))

# train the model
rf_model <- train(as.factor(primary_fuel)~., sampsize=7000, train_data,
                  method = "parRF", ntree=300, tuneGrid = tunegrid,
                  trControl = trControl)

#the loop
loop_rf <- foreach(ntree=rep(300,cores),
                   .combine = combine,
                   .multicombine = TRUE,
                   .packages = "randomForest") %dopar%
  randomForest(as.factor(primary_fuel)~., sampsize =7000, train_data,
               ntree=ntree, tuneGrid = tunegrid)

stopCluster(clt)
print(rf_model)
```

```
## Parallel Random Forest
##
## 23232 samples
##     5 predictor
##    14 classes: '1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '15'
##
## No pre-processing
## Resampling: Cross-Validated (7 fold)
## Summary of sample sizes: 19913, 19914, 19915, 19915, 19912, 19912, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    1    0.8335924  0.8001141
##    2    0.8387999  0.8066858
##    3    0.8410380  0.8094368
##    4    0.8394450  0.8075586
##    5    0.8392735  0.8073395
##    6    0.8400485  0.8082728
##    7    0.8387999  0.8068105
##    8    0.8388859  0.8069214
##    9    0.8402631  0.8085446
##   10    0.8395319  0.8076964
##   11    0.8394462  0.8075604
##   12    0.8384127  0.8063349
##   13    0.8393593  0.8074798
##   14    0.8402639  0.8085322
##   15    0.8398326  0.8080239
##   16    0.8393594  0.8074548
##   17    0.8401773  0.8084285
##   18    0.8394025  0.8074942
##   19    0.8406505  0.8090080
##   20    0.8395750  0.8077139
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 3.
```

From the above analysis the best parameter for mtry is 3. We will repeat our model with this parameter and try to fine tune the sample size and the number of times we do cross validations.

```r
library(randomForest)
library(doParallel)

set.seed(1558)

# set up parallelism
cores <- 3
clt <- makePSOCKcluster(cores)
registerDoParallel(clt)

# set mtry
tunegrid = expand.grid(.mtry = 3)

# choose parameters for training
trControl <- trainControl(method = "cv",
    number = 10,
    search = "grid")

# train the model
```

```
rf_model1 <- train(as.factor(primary_fuel)~., sampsize=10000, train_data,
                   method = "parRF", ntree=500, tuneGrid = tunegrid,
                   trControl = trControl)

#the loop
loop_rf <- foreach(ntree=rep(500,cores),
                   .combine = combine,
                   .multicombine = TRUE,
                   .packages = "randomForest") %dopar%
  randomForest(as.factor(primary_fuel)~., sampsize =10000,train_data, ntree=ntree, tuneGrid = tunegrid)

stopCluster(clt)
print(rf_model1)
```

```
## Parallel Random Forest
##
## 23232 samples
##     5 predictor
##    14 classes: '1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '15'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 20909, 20909, 20910, 20911, 20908, 20905, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.8506807  0.8211542
##
## Tuning parameter 'mtry' was held constant at a value of 3
```

Our accuracy with our second model is 0.85 which is not bad considering our sample size of only 10000 out of the 23000 available samples. We will stop our tuning at this point and check for the prediction power of the model next.

```
set.seed(1558)
# make the prediction
pred <- predict(rf_model1,test_data)

# Check the accuracy of the prediction
acc <- confusionMatrix(pred, as.factor(test_data$primary_fuel))
acc$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##      0.8539983      0.8252438      0.8446591      0.8629807      0.2426483
## AccuracyPValue  McnemarPValue
##      0.0000000            NaN
```

Our prediction parameters give as an accuracy of 0.85 which is good for a first time model to which we have not done extensive parameter optimization due to time and computational cost limitations.

Support Vector Machines

Random Forest is known for over fitting data in its models. We will use Support Vector Machines(SVMs) to try and get a prediction that is more discriminating in its classification. Better classes will lead to better prediction.

We have already split our data and trained it in the previous part. This time we will only train our data and see how well the new model performs.

```r
set.seed(1558)
#train the model
svm_model <- svm(as.factor(primary_fuel)~., train_data,
                 type = "C-classification", kernel = "linear")
print(svm_model)
```

```
##
## Call:
## svm(formula = as.factor(primary_fuel) ~ ., data = train_data, type = "C-classification",
##     kernel = "linear")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  1
##
## Number of Support Vectors:  22093
```

Now let us check the prediction accuracy for our initial svm analysis.

```r
set.seed(1558)
# make the prediction
pred <- predict(svm_model,test_data)

# Check the accuracy of the prediction
acc <- mean(test_data$primary_fuel==pred)
acc
```

```
## [1] 0.3411866
```

Let us increase the cost of our classifier to see if the accuracy will increase from 0.35 which is very bad.

```r
set.seed(1558)
#train the model
svm_model1 <- svm(as.factor(primary_fuel)~., train_data, cost = 20,
                  type = "C-classification", kernel = "linear")
```

```r
set.seed(1558)
# make the prediction
pred <- predict(svm_model1,test_data)

# Check the accuracy of the prediction
acc <- mean(test_data$primary_fuel==pred)
acc
```

```
## [1] 0.3430782
```

Increasing the cost from the default 0f 1 to 20 makes a very small improvement that is negligible. This time we will adjust the type of kernel used and see how better our model gets.

```r
set.seed(1558)
#train the model
svm_model2 <- svm(as.factor(primary_fuel)~., train_data,
                  type = "C-classification", kernel = "radial")
```

```
set.seed(1558)
# make the prediction
pred <- predict(svm_model2,test_data)

# Check the accuracy of the prediction
acc <- mean(test_data$primary_fuel==pred)
acc
```

```
## [1] 0.5795357
```

With radial kernel our prediction becomes a bit better than random chance. We will again try increasing the cost and see if it will have a major improvement or minor one.

```
set.seed(1558)
#train the model
svm_model3 <- svm(as.factor(primary_fuel)~., train_data,cost = 200,
                  type = "C-classification", kernel = "radial")
```

```
set.seed(1558)
# make the prediction
pred <- predict(svm_model3,test_data)

# Check the accuracy of the prediction
acc <- mean(test_data$primary_fuel==pred)
acc
```

```
## [1] 0.6696475
```

In order to achieve at least 70% accuracy our cost of misclassification has to be very high and the decision surface is far from smooth.

3. RESULTS

The initial steps for our analysis involved unsupervised learning in order to try and get a better picture of our data. From the K-means clustering it can be inferred that they are roughly six categories in our data and these categories most likely contain the most prevalent primary fuels in the world. These fuels have the most total energy produced within the span of 6 years as per the database. The fuels are coal,gas,hydro,nuclear,wind and bio-waste. That is to mean that whenever you choose a location in the world and try to build a power plant then these six are the initial sources of energy to consider as their output will be sufficient and their technology is mature.

The next step was to do supervised learning with the most important parameters in our dataset. The goal was to predict primary fuels using the provided parameters which included the longitude and latitude of the powerplants, the energy produced by the plants and the capacity of the powerplant. Random forest models worked best for our analysis and managed a prediction accuracy of 0.85. This is not very good accuracy but it is enough to have informed decisions on the fuel of choice for a given area after the estimated values of the rest of the parameters have been provided.

The support vector machines were not successful at giving a proper prediction. The level of dimensions of the dataset was a bit high for this supervised learning algorithm. We had to let the cost be very high to just get an acceptable accuracy of 70%. Thus the use of this method will require an investment of time and computational resources to try and fine tune the results and it still may not be plug and play for other datasets that are of the similar nature as the current database.

4. CONCLUSION

The methods used in this analysis have several set backs that may add bias to our predictions. The random forest models are usually hindered by the size of the dataset, my prediction was done on a sample size that is almost half the size of the training data. This is because as the number of trees increase the algorithm gets slower and slower making it hard to make predictions in real time. Random Forest can also overfit and fails to discover patterns that are outside the training set. On the other hand SVM regression is good at extrapolation and we used it for the second part of our prediction. We expected better results from SVM but it was the opposite, the accuracy was below 50% and getting it to 70% needed a lot of bending towards the data.SVM is not good with large datasets that dont have clear separation and since we have 15 classes that overlap into mainly 6 as it was shown in K-means clustering then our results were bound to be mediocre at best.

In conclusion random forest came up as the better algorithm for modelling predictions of the dataset. With a few more iterations we can get a model that has an accuracy above 90% which is very good for our purpose. We can therefore predict the primary fuel that is suitable for a given area provided that we have estimated values for the rest of the parameters.