clustering

Stephen Brownsey

25/12/2019

2 step markov chain I guess

This section is going cover two step clustering based on various clustering methods for comparison. Visit data will be combined into one bee result for each orchard, note in principle this is fundementally flawed as the data does depend on previous years and the assumption that independent of the past is false as pesticide last year affects bee population this year as would be expected. In these scenarios each implementation will have a maximum of 8 clusters. Two after pre-bloom, 4 after and 8 after post bloom.

```
load("data")
markov_data <- data_2012 %>%
  select(ends_with(".pre"), ends_with(".blm"), ends_with(".pos"), orchard) %>%
  unique()

#function to generate the agglomaerative clustering
aggl <- function(data){
  agnes(dist(data, method = "euclidian"),
                   diss=TRUE, method = "ward")
}

#function to reduce copy pasting for each node
aggl_c <- function(data, ends, c_num){
  #define temp dataset of this stage by cluster
    temp <- data %>%
    filter(cluster == c_num) %>%
    select(ends_with(ends))

#If statement as if 1 element in dataset it'll error
if(nrow(temp) > 1){
  temp$cluster <- cutree(aggl(temp) , k = 2)

   output <- data %>%
    filter(cluster == c_num) %>%
    select(-cluster) %>%
    mutate(cluster = temp$cluster)
   #Else statement for if 1 element in dataset
  }else{
    #If only 1 element in the dataset -> set the cluster number to 1
    output <- data %>%
      filter(cluster == c_num) %>%
      select(-cluster) %>%
      mutate(cluster = 1)
  }
  output
}


####Agglomerative heirachicale clustering
```

```r
#pre-bloom step
temp <- markov_data %>%
  select(ends_with(".pre"))

temp$cluster <- cutree(aggl(temp) , k = 2)

aggl_node1 <- markov_data %>%
  mutate(cluster = temp$cluster)

#during bloom step 1
aggl_node2 <- aggl_c(aggl_node1, ".blm", 1)

#during bloom step 2
aggl_node3 <- aggl_c(aggl_node1, ".blm", 2)


#post bloom step 1
aggl_node4 <- aggl_c(aggl_node2, ".pos", 1)

#post bloom step 2
aggl_node5 <- aggl_c(aggl_node2, ".pos", 2)

#post bloom step 3
aggl_node6 <- aggl_c(aggl_node3, ".pos", 1)

#post bloom step 4
#Note this node only has 1 element in it so it is automatically set to 1
aggl_node7 <- aggl_c(aggl_node3, ".pos", 2)

orchard_node_agglom <- tibble(orchard = aggl_node4 %>% filter(cluster == 1) %>% select(orchard) %>% as_
  bind_rows(tibble(orchard = aggl_node4 %>% filter(cluster == 2) %>% select(orchard) %>% as_vector, node
  bind_rows(tibble(orchard = aggl_node5 %>% filter(cluster == 1) %>% select(orchard) %>% as_vector, node
  bind_rows(tibble(orchard = aggl_node5 %>% filter(cluster == 2) %>% select(orchard) %>% as_vector, node
  bind_rows(tibble(orchard = aggl_node6 %>% filter(cluster == 1) %>% select(orchard) %>% as_vector, node
  bind_rows(tibble(orchard = aggl_node6 %>% filter(cluster == 2) %>% select(orchard) %>% as_vector, node
  bind_rows(tibble(orchard = aggl_node7 %>% filter(cluster == 1) %>% select(orchard) %>% as_vector, node
```

kmeans approach: Even running the 1001 times it is different everytime: store output later on but run it like 1million times at each stage.

```r
#function to generate the kmeans clustering
k_clustering <- function(data, n = 1001){
#creating the dataset to then calculate the optimal cluster from
  for(i in 1:n){
  if(i == 1){
     k_list <- tibble(kmeans(data, 2)$cluster)
  }else{
     k_list <- bind_cols(k_list, tibble(kmeans(data, 2)$cluster))
  }

  }
  apply(k_list[ ,1:length(k_list)], 1, mfv1) %>%
    enframe(value = "cluster") %>%
    select(cluster)
```

```
}
#pre-bloom step
#Getting the most common clustering

kmeans_c <- function(data, ends, c_num){

  data <- data %>%
    filter(cluster == c_num) %>%
    select(-cluster)

  if(nrow(data) > 2){
    temp <- data  %>%
    select(ends_with(ends))

  cluster <- k_clustering(temp, 2)
 output <- bind_cols(data, cluster)
  }else{
    #Not more than 2 rows and get the error message as:
    #number of cluster centres must lie between 1 and nrow(x)
    output <- data %>%
      mutate(cluster = 1)
  }

 output
}


temp <- k_clustering(markov_data %>% select(-orchard))

kmeans_node1 <- markov_data %>%
  bind_cols(temp)


#during bloom step 1
kmeans_node2 <- kmeans_c(kmeans_node1, ".blm", 1)

#during bloom step 2
kmeans_node3 <- kmeans_c(kmeans_node1, ".blm", 2)


#post bloom step 1
kmeans_node4 <- kmeans_c(kmeans_node2, ".pos", 1)

#post bloom step 2
kmeans_node5 <- kmeans_c(kmeans_node2, ".pos", 2)

#post bloom step 3
kmeans_node6 <- kmeans_c(kmeans_node3, ".pos", 1)

#post bloom step 4
kmeans_node7 <- kmeans_c(kmeans_node3, ".pos", 2)

orchard_node_kmeans <- tibble(orchard = kmeans_node4 %>% filter(cluster == 1) %>% select(orchard) %>% as
```

```
  bind_rows(tibble(orchard = kmeans_node4 %>% filter(cluster == 2) %>% select(orchard) %>% as_vector, no
  bind_rows(tibble(orchard = kmeans_node5 %>% filter(cluster == 1) %>% select(orchard) %>% as_vector, no
  bind_rows(tibble(orchard = kmeans_node5 %>% filter(cluster == 2) %>% select(orchard) %>% as_vector, no
  bind_rows(tibble(orchard = kmeans_node6 %>% filter(cluster == 1) %>% select(orchard) %>% as_vector, no
  bind_rows(tibble(orchard = kmeans_node6 %>% filter(cluster == 2) %>% select(orchard) %>% as_vector, no
  bind_rows(tibble(orchard = kmeans_node7 %>% filter(cluster == 2) %>% select(orchard) %>% as_vector, no
```

Just a look at the difference in the final clusters between the two methods

```
clusterings <- orchard_node_agglom %>%
  arrange(orchard) %>%
  bind_cols(node_km = orchard_node_kmeans %>% arrange(orchard) %>%
              select(node) %>% as_vector)
clusterings
```

```
## # A tibble: 19 x 3
##    orchard  node node_km
##    <fct>   <dbl>   <dbl>
##  1 A           8      10
##  2 B           9      10
##  3 C           9      10
##  4 D           9      10
##  5 E          12      12
##  6 F          13      13
##  7 G          10       8
##  8 H.nooil    11       9
##  9 I           8      10
## 10 J           8       9
## 11 K           8      10
## 12 L           8      10
## 13 M           8      10
## 14 N          14      12
## 15 O.nooil    12      12
## 16 P           8      10
## 17 Q           8       9
## 18 R           8      10
## 19 S          11       9
```

Updating the bee values to match the environments

In this particular analysis, the interest is in the affects of pesticides on bee population. As such, it is necessary to correct, as best we can, for extra factors (confounding?) to make the bee values representative.

All these graphs were plotted as part of EDA by the use of lapply to the a basic plotting function. Here the relationships between some of the main extra factors (are they confounding?) will be examined.

Temperature

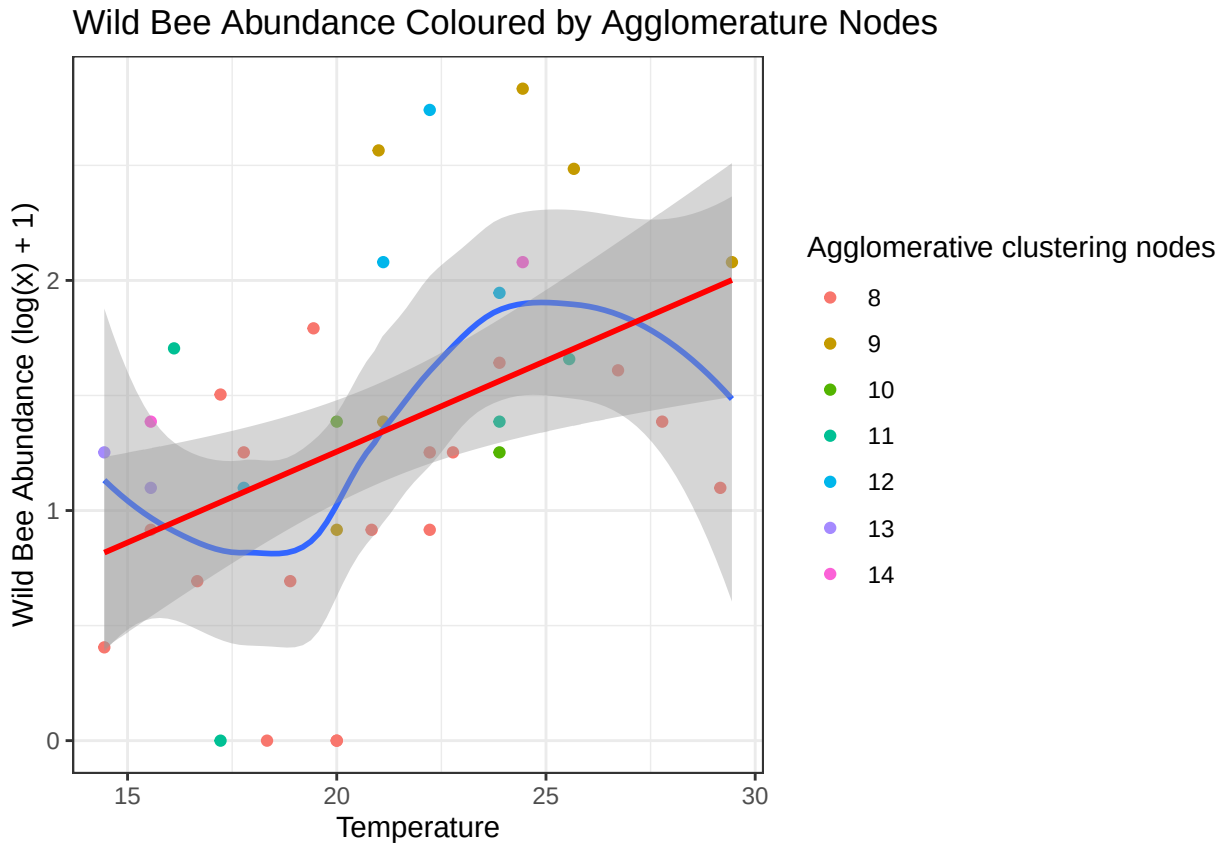
It is known that temperature affects the speed at which bees fly, as such this will have an effect on bee abundance and possibly richness although more bees might not necessarily mean more species are observed. From the previous analysis carried out it is known that a log(x) + 1 transposition of bee count allows for a "better" model fit and as such in general this will the case for our observations.

```
##agglom
wild_bee_abundance_agglom <- data_2012 %>%
  inner_join(orchard_node_agglom) %>%
  ggplot(aes(x = temp, y = log(wildAbF + 1))) +
  geom_point(aes(colour = as_factor(node))) +
  geom_smooth() +
  geom_smooth(method = "lm", colour = "red") +
  labs(x = "Temperature", y = "Wild Bee Abundance (log(x) + 1)",
       title = "Wild Bee Abundance Coloured by Agglomerature Nodes",
       colour = "Agglomerative clustering nodes") +
  theme_bw()
```

```
## Joining, by = "orchard"
```

```
wild_bee_abundance_agglom
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



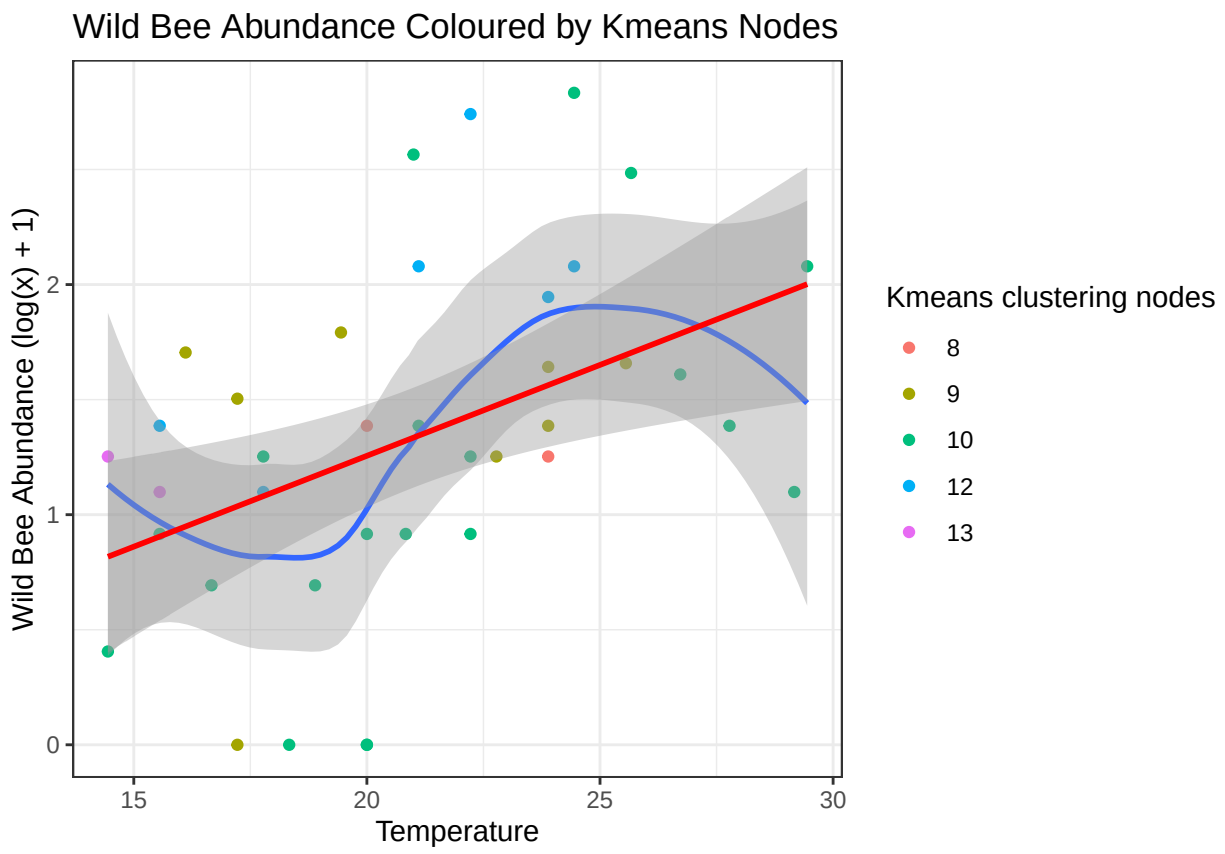

```
##kmeans
wild_bee_abundance_kmeans <- data_2012 %>%
  inner_join(orchard_node_kmeans) %>%
  ggplot(aes(x = temp, y = log(wildAbF + 1))) +
  geom_point(aes(colour = as_factor(node))) +
  geom_smooth() +
  geom_smooth(method = "lm", colour = "red") +
  labs(x = "Temperature", y = "Wild Bee Abundance (log(x) + 1)",
       title = "Wild Bee Abundance Coloured by Kmeans Nodes",
```

```
       colour = "Kmeans clustering nodes") +
  theme_bw()
```

```
## Joining, by = "orchard"
```

```
wild_bee_abundance_kmeans
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



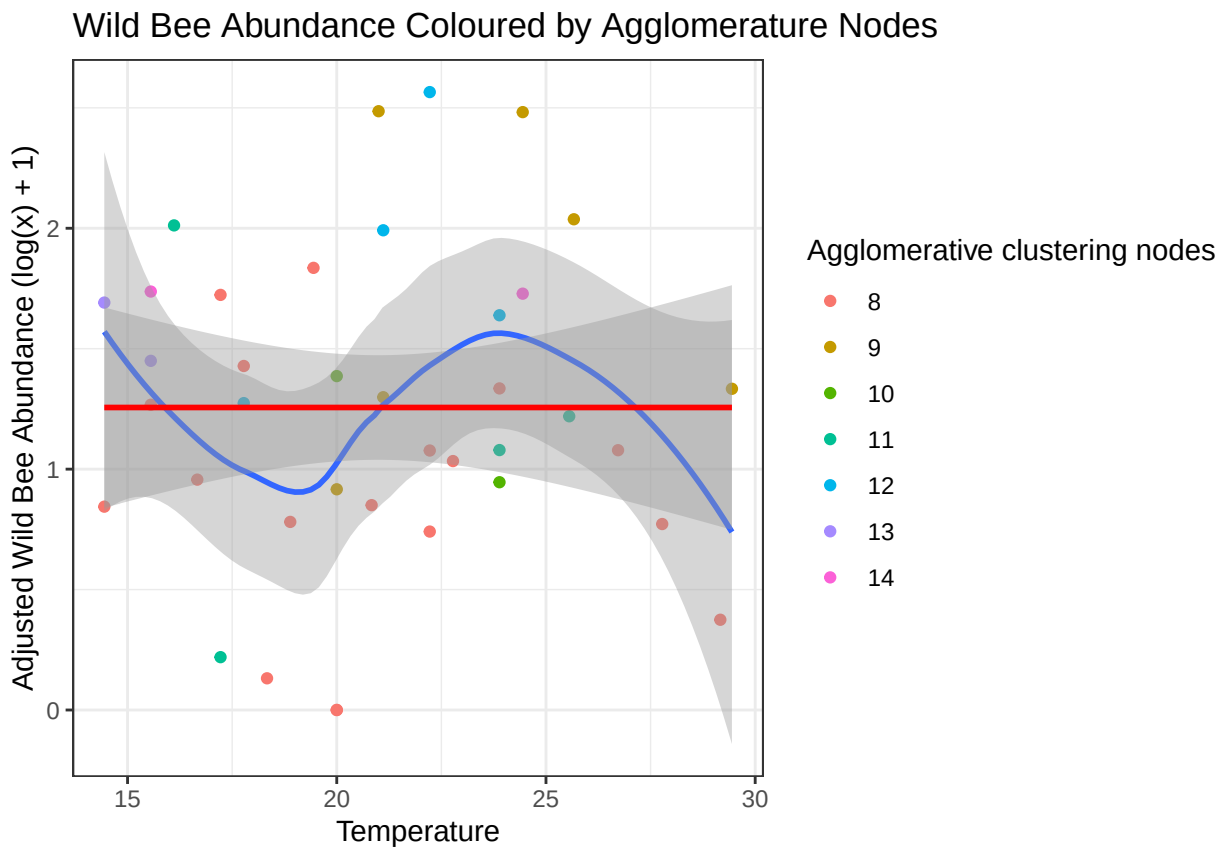

```
##Linear model
lm_temp <- lm(log(data_2012$wildAbF + 1) ~  data_2012$temp)
temp_factor <- tidy(lm_temp) %>%
  slice(2) %>%
  select(estimate) %>%
  as_vector()

#Adjusting bee value as though temp is always 20
adjusted_data <- data_2012 %>%
  inner_join(clusterings) %>%
  mutate(adjusted_bees = (((20 - temp) * temp_factor) + log(wildAbF + 1)))
```

```
## Joining, by = "orchard"
```

```
wild_bee_abundance_agglom_adjusted <- adjusted_data %>%
  ggplot(aes(x = temp, y = adjusted_bees)) +
  geom_point(aes(colour = as_factor(node))) +
  geom_smooth() +
  geom_smooth(method = "lm", colour = "red") +
```

```
  labs(x = "Temperature", y = "Adjusted Wild Bee Abundance (log(x) + 1)",
       title = "Wild Bee Abundance Coloured by Agglomerature Nodes",
       colour = "Agglomerative clustering nodes") +
  theme_bw()
wild_bee_abundance_agglom_adjusted
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



X2000nat

Based on previous research undertaken in the white paper it is known that the X2000nat variable has an effect on the the bee count so it also necessary to adjust for this. Again the value will be adjust as though the X2000nat variable is constant. In this case (X - decide on most appropriate value) will be chosen as the constant value.

Clustering Summarising

```
get_name <- function(x) {
  deparse(substitute(x))
}

#Will need to update data_2012 here if I implement new bee abundance values
#If I come up with multiple scenarios to test I'll just functionalise the below code
bee_values <- data_2012 %>%
```

```
  group_by(orchard) %>%
  summarise(mean_honey_ab = mean(apisAb),
            mean_wild_ab = mean(wildAbF),
            mean_wild_rich = mean(wildRichF)
            )

clusterise <- function(cluster_data, v_name = deparse(substitute(cluster_data))){
  inner_join(bee_values, cluster_data) %>%
    summarise(mean_honey_ab = mean(mean_honey_ab),
              mean_wild_ab = mean(mean_wild_ab),
              mean_wild_rich = mean(mean_wild_rich)) %>%
    mutate(cluster = v_name)
}

agglom_bees <- clusterise(aggl_node1) %>%
  bind_rows(clusterise(aggl_node2)) %>%
  bind_rows(clusterise(aggl_node3)) %>%
  bind_rows(clusterise(aggl_node4)) %>%
  bind_rows(clusterise(aggl_node5)) %>%
  bind_rows(clusterise(aggl_node6)) %>%
  bind_rows(clusterise(aggl_node7)) %>%
  bind_rows(clusterise(aggl_node4 %>% filter(cluster == 1), "agglom_node8")) %>%
  bind_rows(clusterise(aggl_node4 %>% filter(cluster == 2), "agglom_node9")) %>%
  bind_rows(clusterise(aggl_node5 %>% filter(cluster == 1), "agglom_node10")) %>%
  bind_rows(clusterise(aggl_node5 %>% filter(cluster == 2), "agglom_node11")) %>%
  bind_rows(clusterise(aggl_node6 %>% filter(cluster == 1), "agglom_node12")) %>%
  bind_rows(clusterise(aggl_node6 %>% filter(cluster == 2), "agglom_node13")) %>%
  #Only 1 result for cluster as none in the latter cluster
  #Unable to to bind as no common variables for an empty node15
  bind_rows(clusterise(aggl_node7 %>% filter(cluster == 1), "agglom_node14"))
```

```
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
```

```
agglom_bees
```

```
## # A tibble: 14 x 4
##    mean_honey_ab mean_wild_ab mean_wild_rich cluster
##            <dbl>        <dbl>          <dbl> <chr>
##  1          6.56         3.89           2.59 aggl_node1
##  2          6.19         3.46           2.50 aggl_node2
##  3          7.94         5.5            2.94 aggl_node3
```

```
##  4          6.80         3.61           2.54 aggl_node4
##  5          3.75         2.88           2.33 aggl_node5
##  6          8.75         5.67           2.67 aggl_node6
##  7          5.5          5              3.75 aggl_node7
##  8          5.46         2.01           1.77 agglom_node8
##  9         10.8          8.42           4.83 agglom_node9
## 10          3.25         2.75           2.25 agglom_node10
## 11          4            2.94           2.38 agglom_node11
## 12          8            7.38           3.25 agglom_node12
## 13         10.2          2.25           1.5  agglom_node13
## 14          5.5          5              3.75 agglom_node14
```

```
kmeans_bees <- clusterise(kmeans_node1) %>%
  bind_rows(clusterise(kmeans_node2)) %>%
  bind_rows(clusterise(kmeans_node3)) %>%
  bind_rows(clusterise(kmeans_node4)) %>%
  bind_rows(clusterise(kmeans_node5)) %>%
  bind_rows(clusterise(kmeans_node6)) %>%
  bind_rows(clusterise(kmeans_node7)) %>%
  bind_rows(clusterise(kmeans_node4 %>% filter(cluster == 1), "kmeans_node8")) %>%
  bind_rows(clusterise(kmeans_node4 %>% filter(cluster == 2), "kmeans_node9")) %>%
  bind_rows(clusterise(kmeans_node5 %>% filter(cluster == 1), "kmeans_node10")) %>%
  bind_rows(clusterise(kmeans_node5 %>% filter(cluster == 2), "kmeans_node11")) %>%
  bind_rows(clusterise(kmeans_node6 %>% filter(cluster == 1), "kmeans_node12")) %>%
  bind_rows(clusterise(kmeans_node6 %>% filter(cluster == 2), "kmeans_node13")) %>%
  bind_rows(clusterise(kmeans_node7 %>% filter(cluster == 1), "kmeans_node14")) %>%
  bind_rows(clusterise(kmeans_node7 %>% filter(cluster == 2), "kmeans_node15"))
```

```
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
```

```
kmeans_bees
```

```
## # A tibble: 15 x 4
##    mean_honey_ab mean_wild_ab mean_wild_rich cluster
##            <dbl>        <dbl>          <dbl> <chr>
##  1          6.56         3.89           2.59 kmeans_node1
##  2          6.19         3.46           2.50 kmeans_node2
##  3          7.94         5.5            2.94 kmeans_node3
##  4          6.5          3.24           2.72 kmeans_node4
##  5          6.04         3.58           2.39 kmeans_node5
##  6          7.94         5.5            2.94 kmeans_node6
##  7        NaN          NaN            NaN    kmeans_node7
```

```
##  8          3.25         2.75           2.25 kmeans_node8
##  9          7.31         3.36           2.83 kmeans_node9
## 10          6.04         3.58           2.39 kmeans_node10
## 11        NaN          NaN            NaN    kmeans_node11
## 12          7.17         6.58           3.42 kmeans_node12
## 13         10.2          2.25           1.5  kmeans_node13
## 14        NaN          NaN            NaN    kmeans_node14
## 15        NaN          NaN            NaN    kmeans_node15
```

```
#This node not there as no results for it
# %>%
#   bind_rows(clusterise(aggl_node7) %>% filter(cluster == 1)) %>%
#
#   bind_rows(clusterise(aggl_node7) %>% filter(cluster == 2))
```

Will functionalise this later but just for our chat cping it

```
bee_values <- data_2012 %>%
  group_by(orchard) %>%
  summarise(mean_honey_ab = mean(apisAb),
            mean_wild_ab = mean(wildAbF),
            mean_wild_rich = mean(wildRichF)
            )

clusterise <- function(cluster_data, v_name = deparse(substitute(cluster_data))){
  inner_join(bee_values, cluster_data) %>%
    summarise(mean_honey_ab = mean(mean_honey_ab),
              mean_wild_ab = mean(mean_wild_ab),
              mean_wild_rich = mean(mean_wild_rich)) %>%
    mutate(cluster = v_name)
}

agglom_bees <- clusterise(aggl_node1) %>%
  bind_rows(clusterise(aggl_node2)) %>%
  bind_rows(clusterise(aggl_node3)) %>%
  bind_rows(clusterise(aggl_node4)) %>%
  bind_rows(clusterise(aggl_node5)) %>%
  bind_rows(clusterise(aggl_node6)) %>%
  bind_rows(clusterise(aggl_node7)) %>%
  bind_rows(clusterise(aggl_node4 %>% filter(cluster == 1), "agglom_node8")) %>%
  bind_rows(clusterise(aggl_node4 %>% filter(cluster == 2), "agglom_node9")) %>%
  bind_rows(clusterise(aggl_node5 %>% filter(cluster == 1), "agglom_node10")) %>%
  bind_rows(clusterise(aggl_node5 %>% filter(cluster == 2), "agglom_node11")) %>%
  bind_rows(clusterise(aggl_node6 %>% filter(cluster == 1), "agglom_node12")) %>%
  bind_rows(clusterise(aggl_node6 %>% filter(cluster == 2), "agglom_node13")) %>%
  #Only 1 result for cluster as none in the latter cluster
  #Unable to to bind as no common variables for an empty node15
  bind_rows(clusterise(aggl_node7 %>% filter(cluster == 1), "agglom_node14"))
```

```
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
```

```
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
```

```
agglom_bees
```

```
## # A tibble: 14 x 4
##    mean_honey_ab mean_wild_ab mean_wild_rich cluster
##            <dbl>        <dbl>          <dbl> <chr>
##  1          6.56         3.89           2.59 aggl_node1
##  2          6.19         3.46           2.50 aggl_node2
##  3          7.94         5.5            2.94 aggl_node3
##  4          6.80         3.61           2.54 aggl_node4
##  5          3.75         2.88           2.33 aggl_node5
##  6          8.75         5.67           2.67 aggl_node6
##  7          5.5          5              3.75 aggl_node7
##  8          5.46         2.01           1.77 agglom_node8
##  9         10.8          8.42           4.83 agglom_node9
## 10          3.25         2.75           2.25 agglom_node10
## 11          4            2.94           2.38 agglom_node11
## 12          8            7.38           3.25 agglom_node12
## 13         10.2          2.25           1.5  agglom_node13
## 14          5.5          5              3.75 agglom_node14
```

```
kmeans_bees <- clusterise(kmeans_node1) %>%
  bind_rows(clusterise(kmeans_node2)) %>%
  bind_rows(clusterise(kmeans_node3)) %>%
  bind_rows(clusterise(kmeans_node4)) %>%
  bind_rows(clusterise(kmeans_node5)) %>%
  bind_rows(clusterise(kmeans_node6)) %>%
  bind_rows(clusterise(kmeans_node7)) %>%
  bind_rows(clusterise(kmeans_node4 %>% filter(cluster == 1), "kmeans_node8")) %>%
  bind_rows(clusterise(kmeans_node4 %>% filter(cluster == 2), "kmeans_node9")) %>%
  bind_rows(clusterise(kmeans_node5 %>% filter(cluster == 1), "kmeans_node10")) %>%
  bind_rows(clusterise(kmeans_node5 %>% filter(cluster == 2), "kmeans_node11")) %>%
  bind_rows(clusterise(kmeans_node6 %>% filter(cluster == 1), "kmeans_node12")) %>%
  bind_rows(clusterise(kmeans_node6 %>% filter(cluster == 2), "kmeans_node13")) %>%
  bind_rows(clusterise(kmeans_node7 %>% filter(cluster == 1), "kmeans_node14")) %>%
  bind_rows(clusterise(kmeans_node7 %>% filter(cluster == 2), "kmeans_node15"))
```

```
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
```

```
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
## Joining, by = "orchard"
```

```
kmeans_bees
```

```
## # A tibble: 15 x 4
##    mean_honey_ab mean_wild_ab mean_wild_rich cluster
##            <dbl>        <dbl>          <dbl> <chr>
##  1          6.56         3.89           2.59 kmeans_node1
##  2          6.19         3.46           2.50 kmeans_node2
##  3          7.94         5.5            2.94 kmeans_node3
##  4          6.5          3.24           2.72 kmeans_node4
##  5          6.04         3.58           2.39 kmeans_node5
##  6          7.94         5.5            2.94 kmeans_node6
##  7        NaN          NaN            NaN    kmeans_node7
##  8          3.25         2.75           2.25 kmeans_node8
##  9          7.31         3.36           2.83 kmeans_node9
## 10          6.04         3.58           2.39 kmeans_node10
## 11        NaN          NaN            NaN    kmeans_node11
## 12          7.17         6.58           3.42 kmeans_node12
## 13         10.2          2.25           1.5  kmeans_node13
## 14        NaN          NaN            NaN    kmeans_node14
## 15        NaN          NaN            NaN    kmeans_node15
```

```
#This node not there as no results for it
# %>%
#   bind_rows(clusterise(aggl_node7) %>% filter(cluster == 1)) %>%
#
#   bind_rows(clusterise(aggl_node7) %>% filter(cluster == 2))
```