Course: Practical Introduction to Data Science – 2020

Assignment 3: Report

Abstract

In the UK, weather data is collected by 37 weather stations across the country and accumulated over the years. Data on wellbeing is also collected from people who self-report how happy they are over the years. This project uses these two datasets, to perform the following tasks using R and RStudio:

1- Determine whether weather stations have similar weather using a k-means clustering algorithm to find weather clusters
2- Use the k-nearest neighbors (KNN) classification algorithm with weather stations data, to classify the stations as either part of the northern third, central third or southern third of the UK
3- Use the weather stations data together with the self-reported happiness ratings to find our whether there is a correlation between weather and happiness

This project successfully cleaned both datasets and used them to achieve the above-mentioned tasks. The k-means clustering algorithm was implemented and the results showed 3 clusters of weather stations with similar weather. Using the KNN classification algorithm, weather stations were classified into various regions with an accuracy of 60%. Finally, using a spearman rank-based correlation on the weather and happiness data showed a very loose association between weather and happiness or wellbeing.

Data Sources

The two datasets used in this project were gotten from the following locations

1. Historic weather/climate data from the UK – https://www.metoffice.gov.uk/research/climate/maps-and-data/historic-station-data
2. Self-reported happiness statistics from the UK (2014/2015) –

https://www.ons.gov.uk/peoplepopulationandcommunity/wellbeing/datasets/personalwellbeingestimatesgeographicalbreakdown

Methodology, Approach and Tools

The datasets were downloaded from the sources above and stored locally on a pc to be cleaned for use.

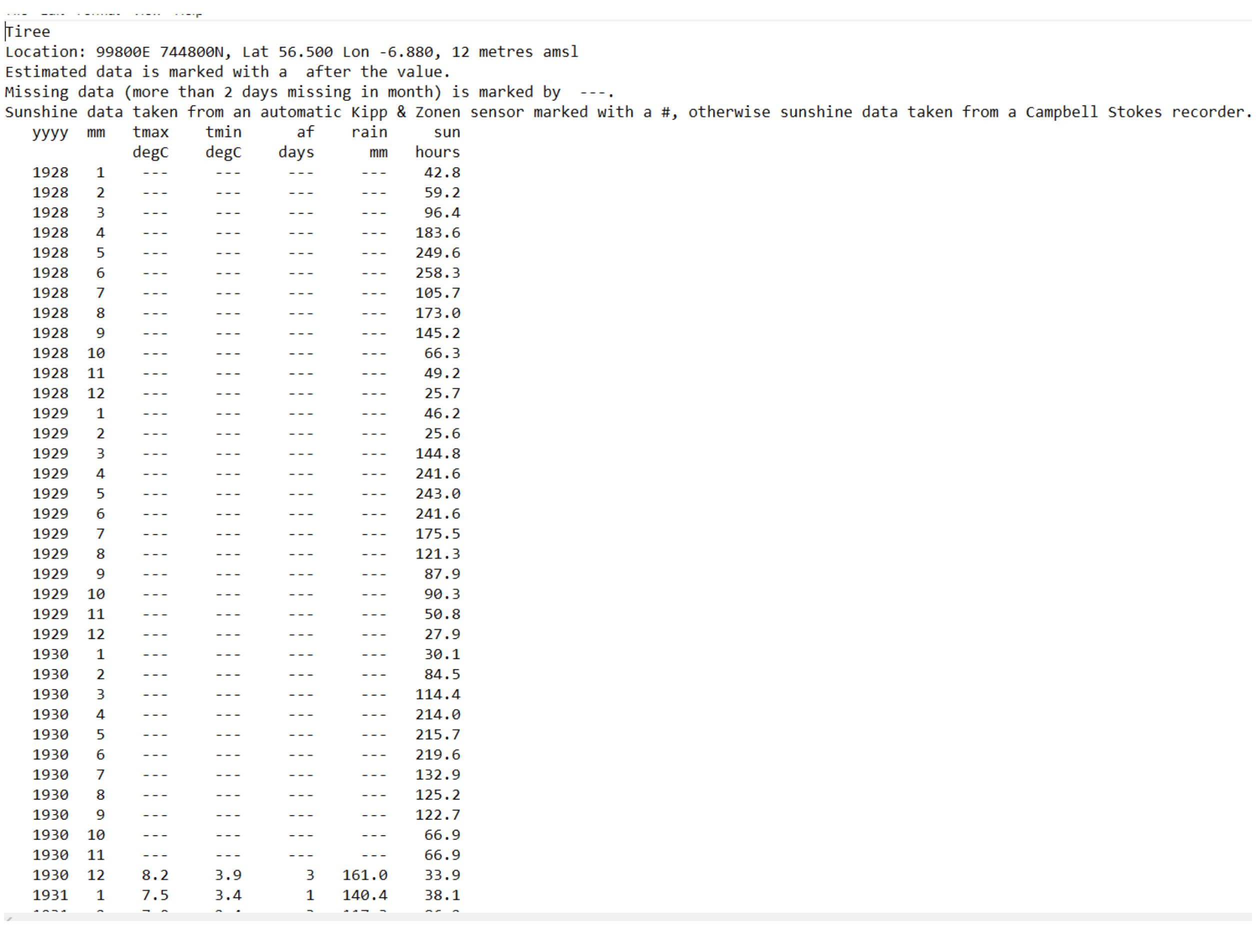
Tools

This project was done in R using RStudio. The curl command line was also used to ease data cleaning.

Weather datasets

The weather data contains 37 datasets corresponding to 37 different weather stations. Each dataset contains short descriptive information about the data and six columns [year, month, maximum temperature (deg Celsius), minimum temperature (deg Celsius), air frost (in days), rainfall (mm) and sunlight duration (hours)], all saved in .txt files.

Some column values contain estimates accompanied by *, and some contains missing values denoted by ---. The sunlight data may have a # appended to it depending on the instrument that was used to measure the values. Some data values are also labelled as “provisional” to indicate that those values haven’t been verified or confirmed.



```
Tiree
Location: 99800E 744800N, Lat 56.500 Lon -6.880, 12 metres amsl
Estimated data is marked with a  after the value.
Missing data (more than 2 days missing in month) is marked by  ---.
Sunshine data taken from an automatic Kipp & Zonen sensor marked with a #, otherwise sunshine data taken from a Campbell Stokes recorder.
   yyyy  mm   tmax    tmin      af    rain     sun
              degC    degC    days      mm   hours
   1928   1    ---     ---     ---     ---    42.8
   1928   2    ---     ---     ---     ---    59.2
   1928   3    ---     ---     ---     ---    96.4
   1928   4    ---     ---     ---     ---   183.6
   1928   5    ---     ---     ---     ---   249.6
   1928   6    ---     ---     ---     ---   258.3
   1928   7    ---     ---     ---     ---   105.7
   1928   8    ---     ---     ---     ---   173.0
   1928   9    ---     ---     ---     ---   145.2
   1928  10    ---     ---     ---     ---    66.3
   1928  11    ---     ---     ---     ---    49.2
   1928  12    ---     ---     ---     ---    25.7
   1929   1    ---     ---     ---     ---    46.2
   1929   2    ---     ---     ---     ---    25.6
   1929   3    ---     ---     ---     ---   144.8
   1929   4    ---     ---     ---     ---   241.6
   1929   5    ---     ---     ---     ---   243.0
   1929   6    ---     ---     ---     ---   241.6
   1929   7    ---     ---     ---     ---   175.5
   1929   8    ---     ---     ---     ---   121.3
   1929   9    ---     ---     ---     ---    87.9
   1929  10    ---     ---     ---     ---    90.3
   1929  11    ---     ---     ---     ---    50.8
   1929  12    ---     ---     ---     ---    27.9
   1930   1    ---     ---     ---     ---    30.1
   1930   2    ---     ---     ---     ---    84.5
   1930   3    ---     ---     ---     ---   114.4
   1930   4    ---     ---     ---     ---   214.0
   1930   5    ---     ---     ---     ---   215.7
   1930   6    ---     ---     ---     ---   219.6
   1930   7    ---     ---     ---     ---   132.9
   1930   8    ---     ---     ---     ---   125.2
   1930   9    ---     ---     ---     ---   122.7
   1930  10    ---     ---     ---     ---    66.9
   1930  11    ---     ---     ---     ---    66.9
   1930  12    8.2     3.9       3   161.0    33.9
   1931   1    7.5     3.4       1   140.4    38.1
```

Figure 1: Snippet of sample weather dataset

For this project, only the temperature and rainfall data were used as weather instruments. This is because, temperature and rainfall are the most significant determinants of the weather of a given location. Air frost days and sunlight data are both positively correlated to temperature and can be left out.

Wellbeing dataset (Happiness Dataset)

The happiness dataset is a huge one with a lot of information. From this dataset, the happiness ratings (average (mean) ratings) for 12 regions of interest were manually copied from the excel file and saved in a .txt file alongside the names of the regions, their latitudes, and their longitudes.

Data Cleaning

For all datasets to be used for any analysis, they had to be cleaned to get technically correct and consistent data.

Before loading the weather datasets into R, all occurrences of the character * were removed from the datasets using the curl command “sed 's/*//g' file_name” in the curl command line and passing the name of each dataset to it.

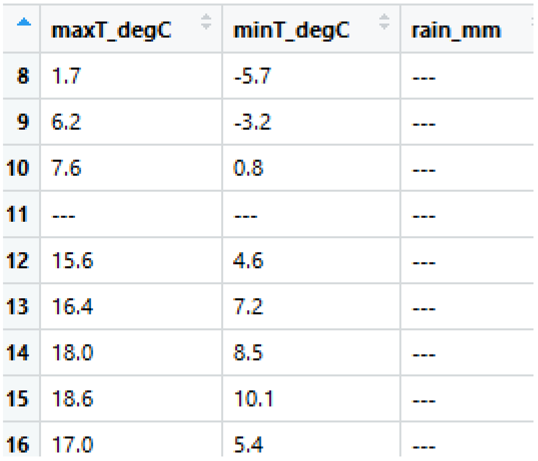
Starting with the weather datasets, a for-loop was created in R to load all 37 .txt files containing the datasets into RStudio and looped through them to do the cleaning. Each dataset was read into a csv and converted into a data frame for easy processing.

```
for (file in files){
  fileDir = paste("dataset1/",file,sep = "")
  fileData <- read.csv(fileDir, header = FALSE, sep="",dec=".") #read each file into a csv to ease cleaning
```

The background information about the dataset was removed from the first 7 or 8 rows. Also, the columns for, year, month, air frost and sunlight were also removed from the dataset and the remaining columns appropriately renamed.

```
# remove extra info on first 7 lines and remove airfrost and sunshine data, and month, year
data_rightRowsAndColumns<-stationData[-c(1:7),c(3,4,6)]

#rename the columns.
names(data_rightRowsAndColumns)<-c("maxT_degC", "minT_degC", "rain_mm")
```

Assumption: It is assumed that the weather conditions for each station over the years does not change too much. Therefore, the year and month column can be removed leaving only the columns with weather instruments.



	maxT_degC	minT_degC	rain_mm
8	1.7	-5.7	---
9	6.2	-3.2	---
10	7.6	0.8	---
11	---	---	---
12	15.6	4.6	---
13	16.4	7.2	---
14	18.0	8.5	---
15	18.6	10.1	---
16	17.0	5.4	---

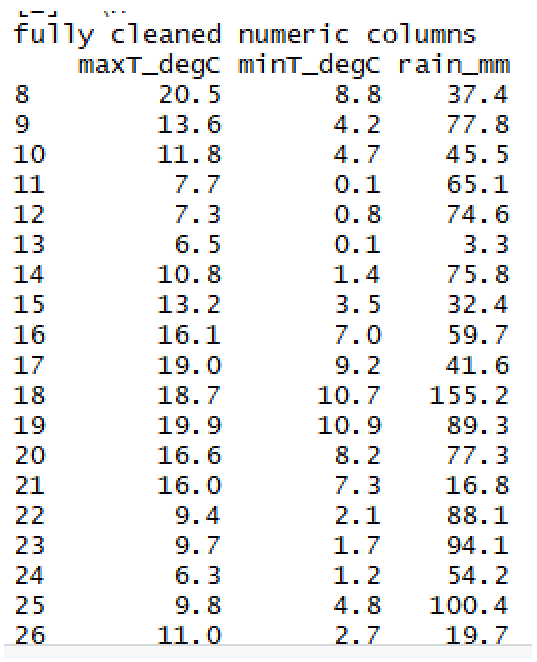
Figure 2: Weather dataset snippet with only columns of interest

All rows containing missing values denoted by “---" were removed to ensure that we only work with complete and consistent observations.

```
#remove rows with missing records (denoted by ---)
dataWithNoMissingValues <- data_rightRowsAndColumns[!(data_rightRowsAndColumns$maxT_degC =="---" |
                                                      data_rightRowsAndColumns$minT_degC =="---" |
                                                      data_rightRowsAndColumns$rain_mm =="---"), ]
```

All columns in the dataset were converted into numeric values to enable calculations such as the mean for each column to be calculated. A final check for empty cells was done to ensure that all datasets were clean, correct, and consistent.

```
#Make all columns numeric
indx <- sapply(dataWithNoMissingValues, is.factor)
dataWithNoMissingValues[indx] <- lapply(dataWithNoMissingValues[indx], function(x) as.numeric(as.character(x)))

dataFrames<-na.omit(dataWithNoMissingValues)  #do final check and remove any NA values
```



```
fully cleaned numeric columns
    maxT_degC minT_degC rain_mm
8        20.5       8.8    37.4
9        13.6       4.2    77.8
10       11.8       4.7    45.5
11        7.7       0.1    65.1
12        7.3       0.8    74.6
13        6.5       0.1     3.3
14       10.8       1.4    75.8
15       13.2       3.5    32.4
16       16.1       7.0    59.7
17       19.0       9.2    41.6
18       18.7      10.7   155.2
19       19.9      10.9    89.3
20       16.6       8.2    77.3
21       16.0       7.3    16.8
22        9.4       2.1    88.1
23        9.7       1.7    94.1
24        6.3       1.2    54.2
25        9.8       4.8   100.4
26       11.0       2.7    19.7
```

Figure 3: Snippet of Clean and consistent dataset

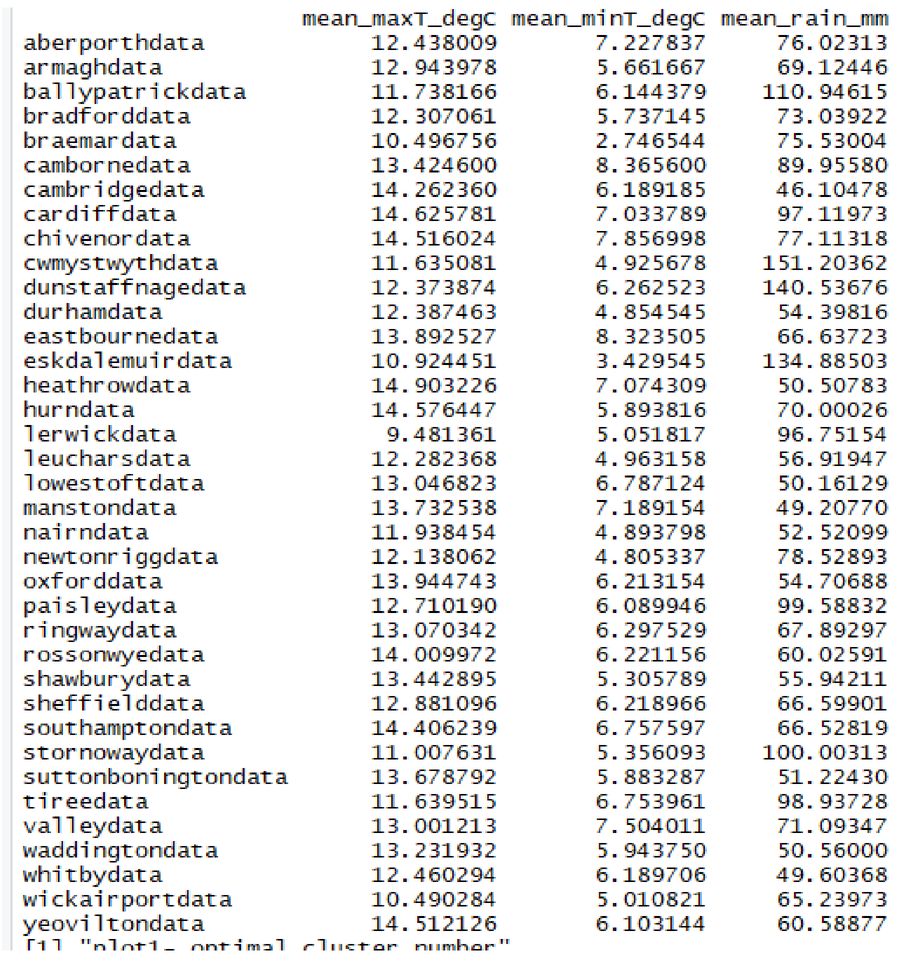
Assignment Part 1: Clustering using k-means

In this section, a clustering algorithm is used to see if the weather stations can be placed into clusters with similar weather.

To run a clustering algorithm on the weather station data, all observations in each dataset were used. The mean values for each column in each dataset were calculated and returned as a data frame. **Here, there is an assumption that climatic conditions have not changed too much over the years and that the mean values for temperature and rainfall for all the years, would give a better estimate for the weather at each station at any given time.**

```
getMeanOfColumns<-function(cleanDataFrame){

  cmeans<-colMeans(cleanDataFrame,na.rm=TRUE)

  #mean for temp and rainfall data for this dataset

  return(cmeans)
}
```

Putting together the mean values for all columns in each dataset, gives the following



```
                     mean_maxT_degC mean_minT_degC mean_rain_mm
aberporthdata             12.438009       7.227837     76.02313
armaghdata                12.943978       5.661667     69.12446
ballypatrickdata          11.738166       6.144379    110.94615
bradforddata              12.307061       5.737145     73.03922
braemardata               10.496756       2.746544     75.53004
cambornedata              13.424600       8.365600     89.95580
cambridgedata             14.262360       6.189185     46.10478
cardiffdata               14.625781       7.033789     97.11973
chivenordata              14.516024       7.856998     77.11318
cwmystwythdata            11.635081       4.925678    151.20362
dunstaffnagedata          12.373874       6.262523    140.53676
durhamdata                12.387463       4.854545     54.39816
eastbournedata            13.892527       8.323505     66.63723
eskdalemuirdata           10.924451       3.429545    134.88503
heathrowdata              14.903226       7.074309     50.50783
hurndata                  14.576447       5.893816     70.00026
lerwickdata                9.481361       5.051817     96.75154
leucharsdata              12.282368       4.963158     56.91947
lowestoftdata             13.046823       6.787124     50.16129
manstondata               13.732538       7.189154     49.20770
nairndata                 11.938454       4.893798     52.52099
newtonriggdata            12.138062       4.805337     78.52893
oxforddata                13.944743       6.213154     54.70688
paisleydata               12.710190       6.089946     99.58832
ringwaydata               13.070342       6.297529     67.89297
rossonwyedata             14.009972       6.221156     60.02591
shawburydata              13.442895       5.305789     55.94211
sheffielddata             12.881096       6.218966     66.59901
southamptondata           14.406239       6.757597     66.52819
stornowaydata             11.007631       5.356093    100.00313
suttonboningtondata       13.678792       5.883287     51.22430
tireedata                 11.639515       6.753961     98.93728
valleydata                13.001213       7.504011     71.09347
waddingtondata            13.231932       5.943750     50.56000
whitbydata                12.460294       6.189706     49.60368
wickairportdata           10.490284       5.010821     65.23973
yeoviltondata             14.512126       6.103144     60.58877
[1] "plot1- optimal cluster number"
```

Figure 4: Dataset with row-names and mean values for each column for each weather station

This data can now be used in a k-means clustering algorithm. The k-means algorithm is used here because of its simplicity and accuracy in generalizing clusters of different size and shapes in datasets. Also, because the dimensions of the datasets are not large, k-means can do a better clustering job.

To use the dataset in figure 4, the row-names need to be dropped. The data was also scaled or normalized to carter for the difference in scales for the different columns. This helps to improve the convergence of the k-means algorithm. For example, temperature values are very small compared to rainfall values due to difference in scales. If the data is not normalized, the clustering will put too much weight on the rainfall data, introducing bias on the output.

```
#remove row names
rownames(cleanedData)<-NULL

#scale (normalise) data to improve convergence for kmeans clustering algorithm
cleanedData.scaled<-scale(cleanedData)

#To determine the optimal number of clusters to use for kmeans, the fviz_nbclust function from factoextra library
print("plot1- optimal cluster number")
plot1<-fviz_nbclust(results.scaled, kmeans, method = "wss") + geom_vline(xintercept = 3, linetype = 2)
```

The k-means algorithm requires that we have an idea of how many clusters to expect (manually choose the value of k). To determine the optimal value of k for the given dataset, a library called factoextra was used with the elbow method (that calculates the minimum within-cluster sum of square (WSS)) to determine the optimal value for k, which was 3.

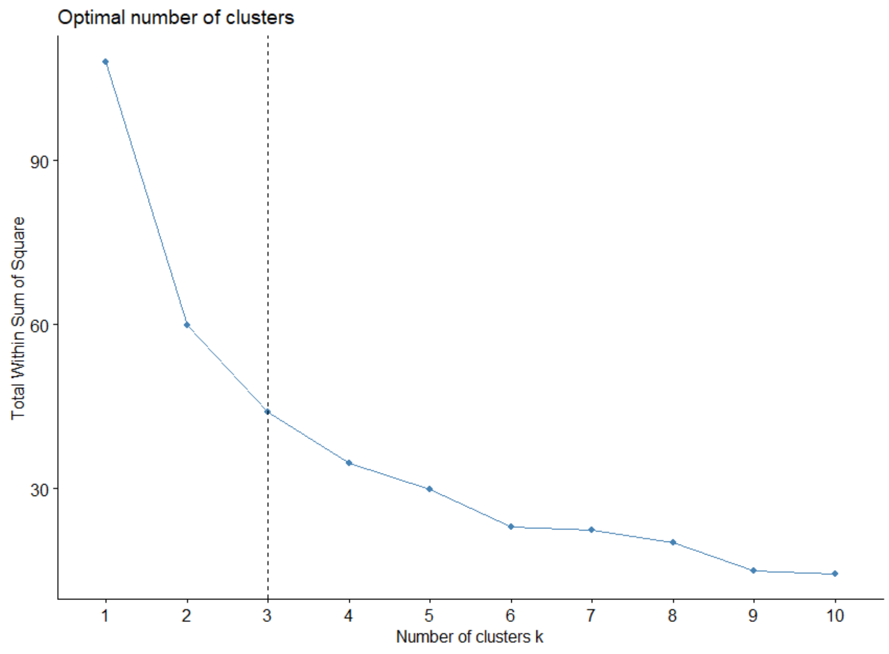
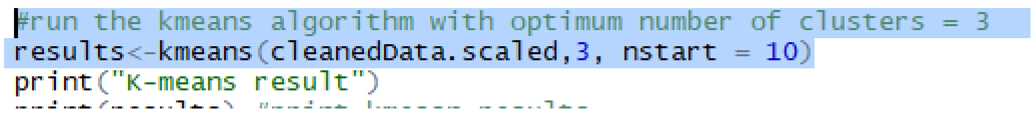


Figure 5: Plot of number of clusters determined from elbow method



```
#run the kmeans algorithm with optimum number of clusters = 3
results<-kmeans(cleanedData.scaled,3, nstart = 10)
print("K-means result")
```

Using the optimal value for k, the kmeans function in R was used to run the k-means clustering.

Results

The k-means algorithm produced 3 clusters of weather stations with similar weather. Figure 5 shows the plot of the different clusters

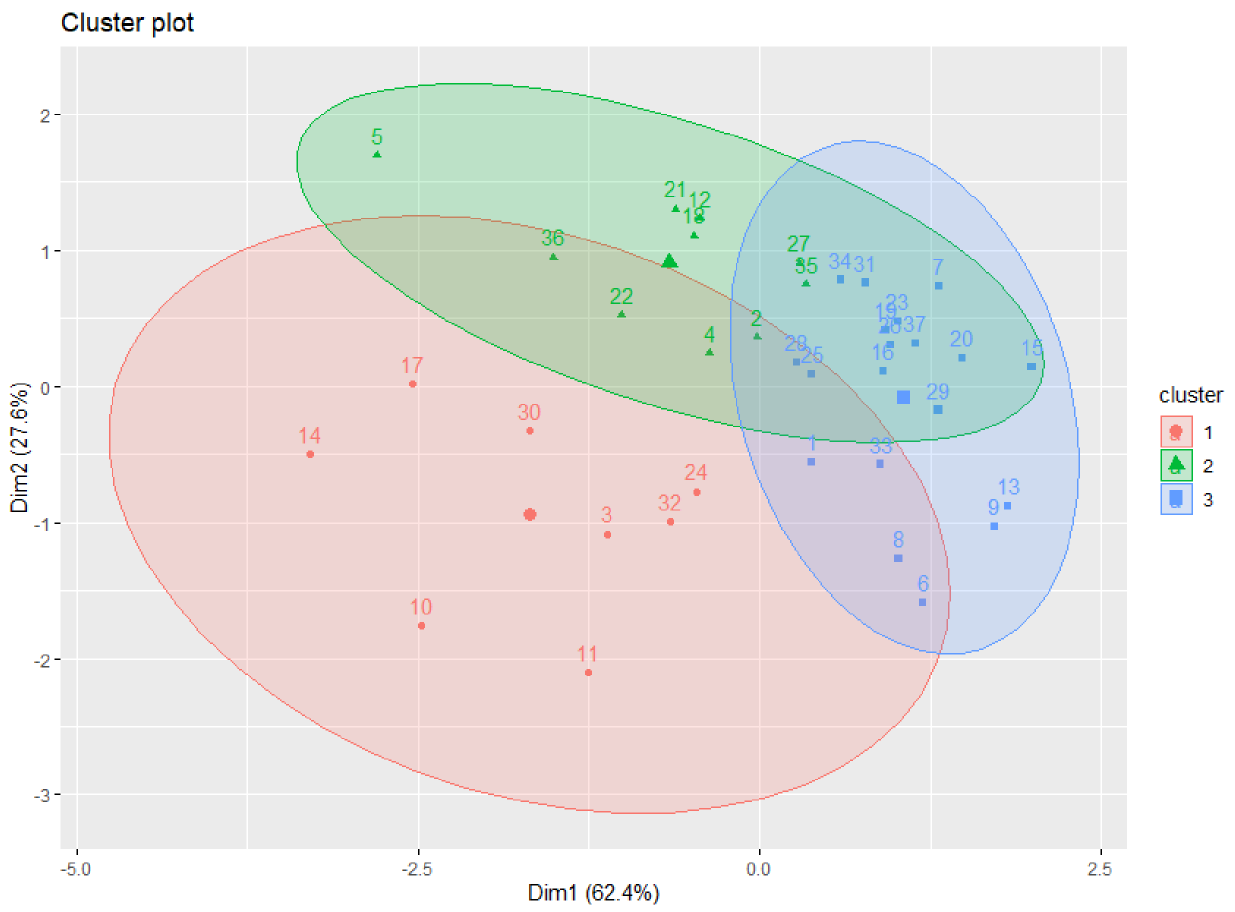
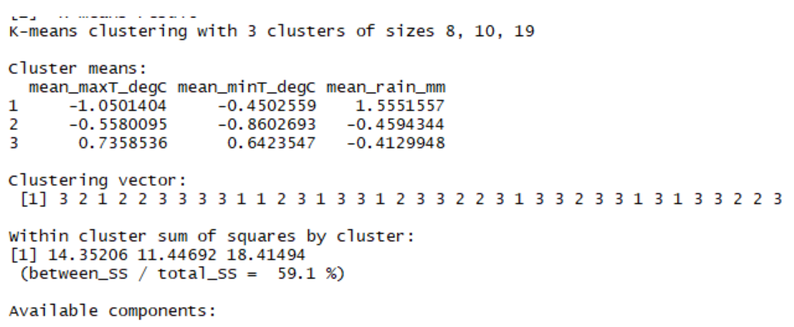


Figure 6: Clusters of weather stations with similar weather

The k-means cluster summary shows that the sizes of clusters 1,2 and 3 and 8,10 and 19, respectively. The rest of the summary is as shown below.

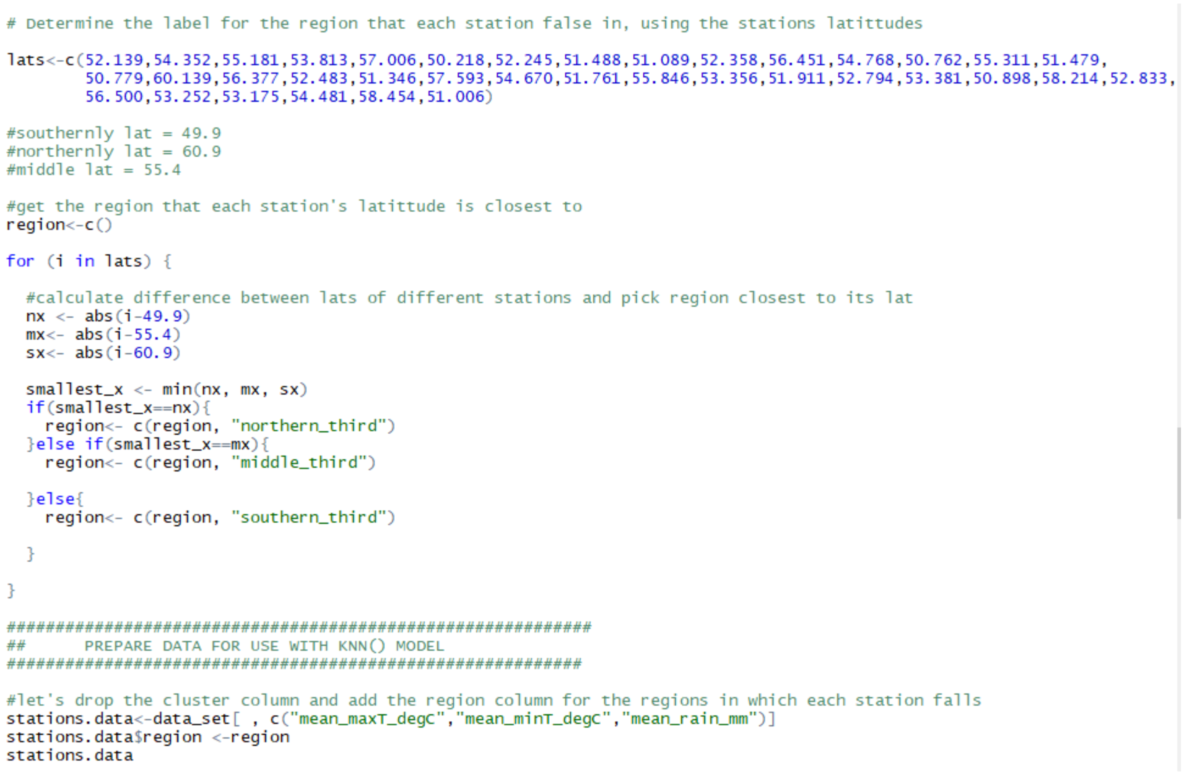


```
K-means clustering with 3 clusters of sizes 8, 10, 19

Cluster means:
  mean_maxT_degC mean_minT_degC mean_rain_mm
1     -1.0501404     -0.4502559    1.5551557
2     -0.5580095     -0.8602693   -0.4594344
3      0.7358536      0.6423547   -0.4129948

Clustering vector:
 [1] 3 2 1 2 2 3 3 3 3 1 1 2 3 1 3 3 1 2 3 3 2 2 2 3 1 3 3 2 3 3 1 3 1 3 3 2 2 3

Within cluster sum of squares by cluster:
[1] 14.35206 11.44692 18.41494
 (between_SS / total_SS =  59.1 %)

Available components:
```

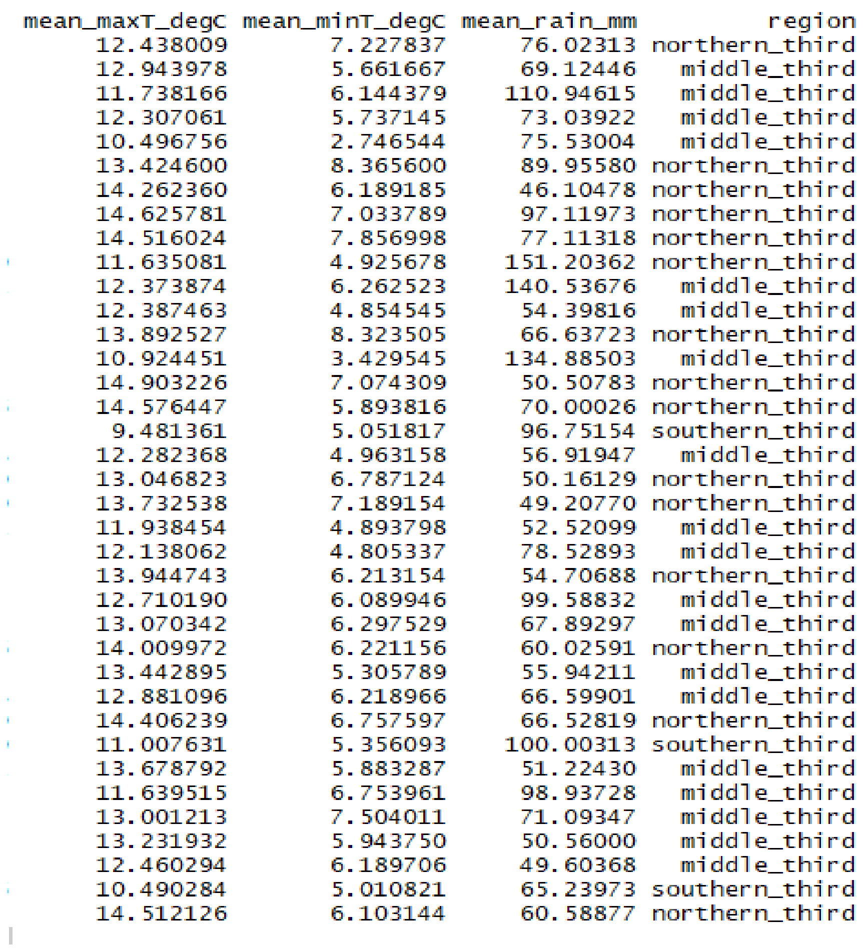
Assignment part 2: Classification using KNN

In this section, the KNN classification model is trained to classify various weather stations into the regions of the country in which they fall, using only their weather data.

To run a classification training on the cleaned dataset, the labels had to be attached to the dataset already prepared as shown in figure 4 above. The value for the latitude for each weather station were manually extracted from each dataset and put in a data frame in R. Using the values for the northerly latitude and southerly latitude of the UK, a middle latitude was obtained as a median of the two. Each station was assigned to the region of the country to whom the difference between their latitudes was the smallest. The code snippet below was used to accomplish this task and the output data frame is shown in figure 7.



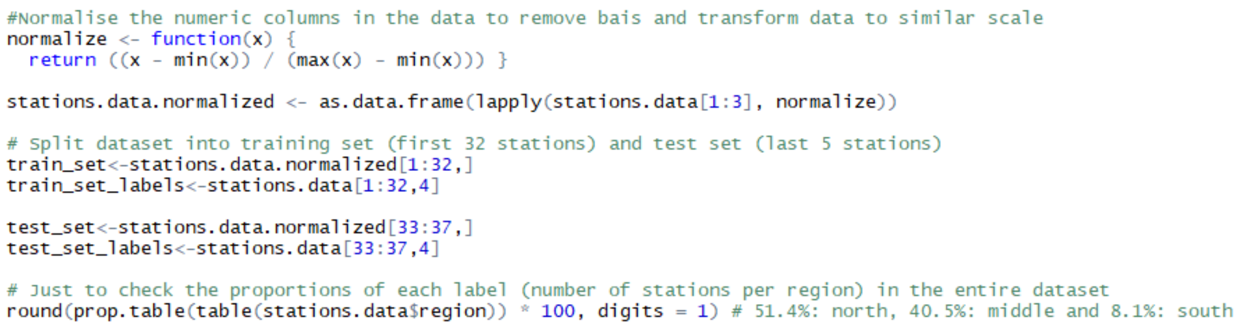
```r
# Determine the label for the region that each station false in, using the stations latittudes

lats<-c(52.139,54.352,55.181,53.813,57.006,50.218,52.245,51.488,51.089,52.358,56.451,54.768,50.762,55.311,51.479,
        50.779,60.139,56.377,52.483,51.346,57.593,54.670,51.761,55.846,53.356,51.911,52.794,53.381,50.898,58.214,52.833,
        56.500,53.252,53.175,54.481,58.454,51.006)

#southernly lat = 49.9
#northernly lat = 60.9
#middle lat = 55.4

#get the region that each station's latittude is closest to
region<-c()

for (i in lats) {

  #calculate difference between lats of different stations and pick region closest to its lat
  nx <- abs(i-49.9)
  mx<- abs(i-55.4)
  sx<- abs(i-60.9)

  smallest_x <- min(nx, mx, sx)
  if(smallest_x==nx){
    region<- c(region, "northern_third")
  }else if(smallest_x==mx){
    region<- c(region, "middle_third")

  }else{
    region<- c(region, "southern_third")

  }

}

###############################################################
##      PREPARE DATA FOR USE WITH KNN() MODEL
##############################################################

#let's drop the cluster column and add the region column for the regions in which each station falls
stations.data<-data_set[ , c("mean_maxT_degC","mean_minT_degC","mean_rain_mm")]
stations.data$region <-region
stations.data
```



```
 mean_maxT_degC mean_minT_degC mean_rain_mm         region
      12.438009       7.227837     76.02313 northern_third
      12.943978       5.661667     69.12446   middle_third
      11.738166       6.144379    110.94615   middle_third
      12.307061       5.737145     73.03922   middle_third
      10.496756       2.746544     75.53004   middle_third
      13.424600       8.365600     89.95580 northern_third
      14.262360       6.189185     46.10478 northern_third
      14.625781       7.033789     97.11973 northern_third
      14.516024       7.856998     77.11318 northern_third
      11.635081       4.925678    151.20362 northern_third
      12.373874       6.262523    140.53676   middle_third
      12.387463       4.854545     54.39816   middle_third
      13.892527       8.323505     66.63723 northern_third
      10.924451       3.429545    134.88503   middle_third
      14.903226       7.074309     50.50783 northern_third
      14.576447       5.893816     70.00026 northern_third
       9.481361       5.051817     96.75154 southern_third
      12.282368       4.963158     56.91947   middle_third
      13.046823       6.787124     50.16129 northern_third
      13.732538       7.189154     49.20770 northern_third
      11.938454       4.893798     52.52099   middle_third
      12.138062       4.805337     78.52893   middle_third
      13.944743       6.213154     54.70688 northern_third
      12.710190       6.089946     99.58832   middle_third
      13.070342       6.297529     67.89297   middle_third
      14.009972       6.221156     60.02591 northern_third
      13.442895       5.305789     55.94211   middle_third
      12.881096       6.218966     66.59901   middle_third
      14.406239       6.757597     66.52819 northern_third
      11.007631       5.356093    100.00313 southern_third
      13.678792       5.883287     51.22430   middle_third
      11.639515       6.753961     98.93728   middle_third
      13.001213       7.504011     71.09347   middle_third
      13.231932       5.943750     50.56000   middle_third
      12.460294       6.189706     49.60368   middle_third
      10.490284       5.010821     65.23973 southern_third
      14.512126       6.103144     60.58877 northern_third
```

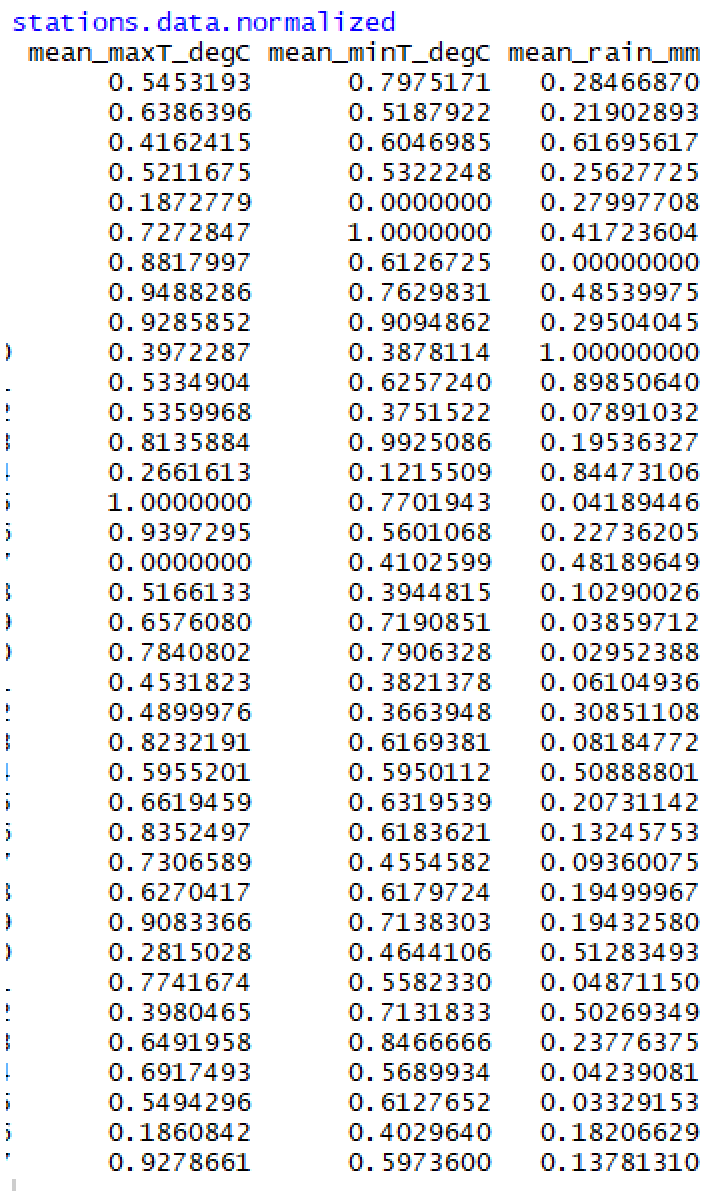
Figure 7: Weather station data with regional labels

A KNN model was implemented and trained to classify weather stations into the corresponding regions (using only weather data in figure 7). Again, before using the data frame in figure 7 to train the model, all numeric column values were normalised (to transform column values to similar scale and remove bias) before the dataset was split into a training set (32 stations) and a test set (5 stations).



```
#Normalise the numeric columns in the data to remove bais and transform data to similar scale
normalize <- function(x) {
  return ((x - min(x)) / (max(x) - min(x))) }

stations.data.normalized <- as.data.frame(lapply(stations.data[1:3], normalize))

# Split dataset into training set (first 32 stations) and test set (last 5 stations)
train_set<-stations.data.normalized[1:32,]
train_set_labels<-stations.data[1:32,4]

test_set<-stations.data.normalized[33:37,]
test_set_labels<-stations.data[33:37,4]

# Just to check the proportions of each label (number of stations per region) in the entire dataset
round(prop.table(table(stations.data$region)) * 100, digits = 1) # 51.4%: north, 40.5%: middle and 8.1%: south
```

Exploring the dataset and looking at the distribution of stations in the various regions of the country, it is seen that 51.4%, 40.5% and 8.1% of the weather stations lie in the northern third, central third and southern third of the country, respectively.

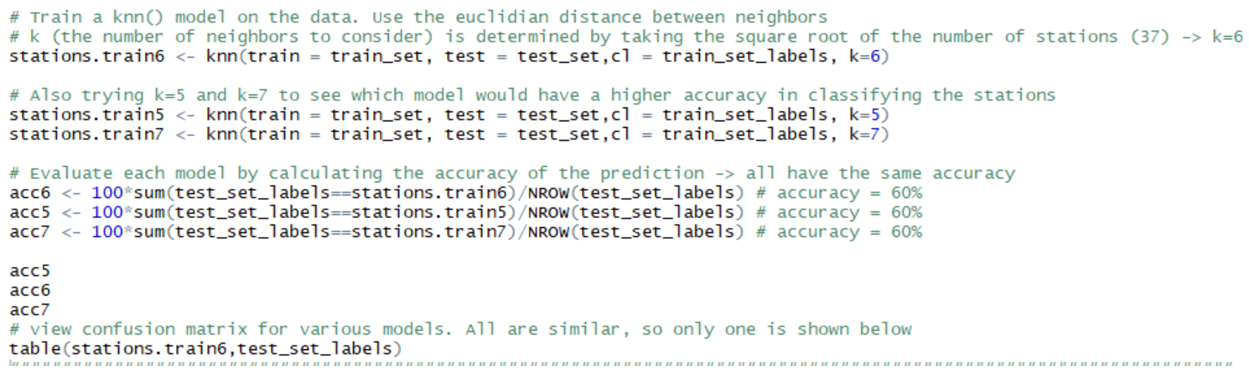


```
stations.data.normalized
 mean_maxT_degC mean_minT_degC mean_rain_mm
      0.5453193      0.7975171   0.28466870
      0.6386396      0.5187922   0.21902893
      0.4162415      0.6046985   0.61695617
      0.5211675      0.5322248   0.25627725
      0.1872779      0.0000000   0.27997708
      0.7272847      1.0000000   0.41723604
      0.8817997      0.6126725   0.00000000
      0.9488286      0.7629831   0.48539975
      0.9285852      0.9094862   0.29504045
      0.3972287      0.3878114   1.00000000
      0.5334904      0.6257240   0.89850640
      0.5359968      0.3751522   0.07891032
      0.8135884      0.9925086   0.19536327
      0.2661613      0.1215509   0.84473106
      1.0000000      0.7701943   0.04189446
      0.9397295      0.5601068   0.22736205
      0.0000000      0.4102599   0.48189649
      0.5166133      0.3944815   0.10290026
      0.6576080      0.7190851   0.03859712
      0.7840802      0.7906328   0.02952388
      0.4531823      0.3821378   0.06104936
      0.4899976      0.3663948   0.30851108
      0.8232191      0.6169381   0.08184772
      0.5955201      0.5950112   0.50888801
      0.6619459      0.6319539   0.20731142
      0.8352497      0.6183621   0.13245753
      0.7306589      0.4554582   0.09360075
      0.6270417      0.6179724   0.19499967
      0.9083366      0.7138303   0.19432580
      0.2815028      0.4644106   0.51283493
      0.7741674      0.5582330   0.04871150
      0.3980465      0.7131833   0.50269349
      0.6491958      0.8466666   0.23776375
      0.6917493      0.5689934   0.04239081
      0.5494296      0.6127652   0.03329153
      0.1860842      0.4029640   0.18206629
      0.9278661      0.5973600   0.13781310
```

Figure 8: Normalized numerical columns of data frame ready for use in KNN model

Although a KNN model is lazy (because it stores the dataset and label verbatim and actually does not do a training with the dataset), it is used here because it is fast and because the dataset is small and there are no concerns for memory size. Also, it has a very easy implementation.

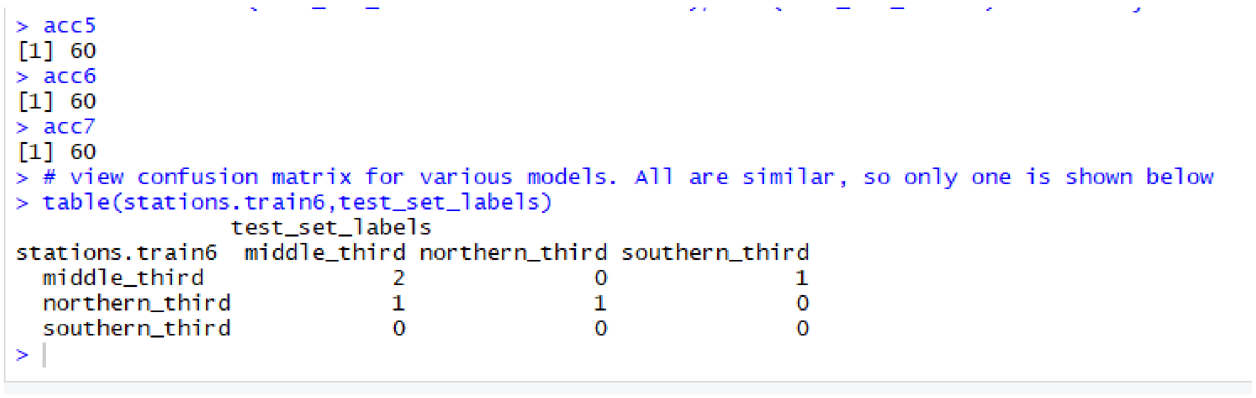
The KNN model in R requires a predetermined value for k (the number of nearest neighbors to consider). This value was determined by taking the square root of total number of weather stations in the entire dataset (giving k=6). However, the model was also trained with k=5 and k=7 and the accuracies for the different models were calculated to determine the model with the most accurate classification. Note that the KNN function uses the Euclidean distance between the neighboring points to determine which class, a test point if closest to.



```
# Train a knn() model on the data. Use the euclidian distance between neighbors
# k (the number of neighbors to consider) is determined by taking the square root of the number of stations (37) -> k=6
stations.train6 <- knn(train = train_set, test = test_set,cl = train_set_labels, k=6)

# Also trying k=5 and k=7 to see which model would have a higher accuracy in classifying the stations
stations.train5 <- knn(train = train_set, test = test_set,cl = train_set_labels, k=5)
stations.train7 <- knn(train = train_set, test = test_set,cl = train_set_labels, k=7)

# Evaluate each model by calculating the accuracy of the prediction -> all have the same accuracy
acc6 <- 100*sum(test_set_labels==stations.train6)/NROW(test_set_labels) # accuracy = 60%
acc5 <- 100*sum(test_set_labels==stations.train5)/NROW(test_set_labels) # accuracy = 60%
acc7 <- 100*sum(test_set_labels==stations.train7)/NROW(test_set_labels) # accuracy = 60%

acc5
acc6
acc7
# view confusion matrix for various models. All are similar, so only one is shown below
table(stations.train6,test_set_labels)
```

Results

All three models with different k values produced the same accuracy of 60%. No station was successfully classified to be in the southern third of the country. The reason behind this is clear. Southerly third stations made up only 8.1% of the entire dataset, so the model have enough data on stations in the southern third region to learn from in order to accurately classify them. Also, in the test set, there is only one southerly weather station, which makes it hard for the model to predict. The model’s confusion matrix is shown below.



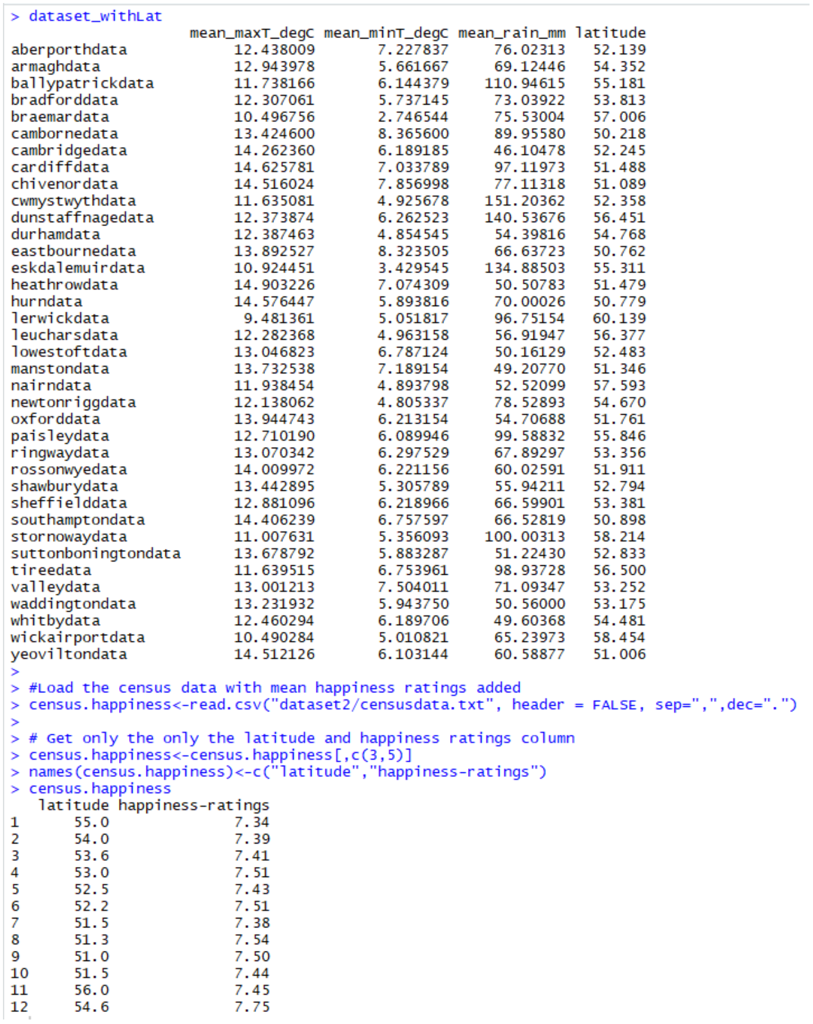
```
> acc5
[1] 60
> acc6
[1] 60
> acc7
[1] 60
> # view confusion matrix for various models. All are similar, so only one is shown below
> table(stations.train6,test_set_labels)
                test_set_labels
stations.train6  middle_third northern_third southern_third
  middle_third              2              0              1
  northern_third            1              1              0
  southern_third            0              0              0
> 
```

Out of 5 stations, the model could only classify 3 of them correctly. This is not a bad accuracy given the limited amount of data the model had to learn from.

Assignment part 3: Correlation between Weather and Happiness

This section seeks to find out whether the weather of a place affects how happy people may be in that region. Here, the cleaned dataset prepared in Part-1 is used, but this time, with the happiness ratings from the 12 census areas.

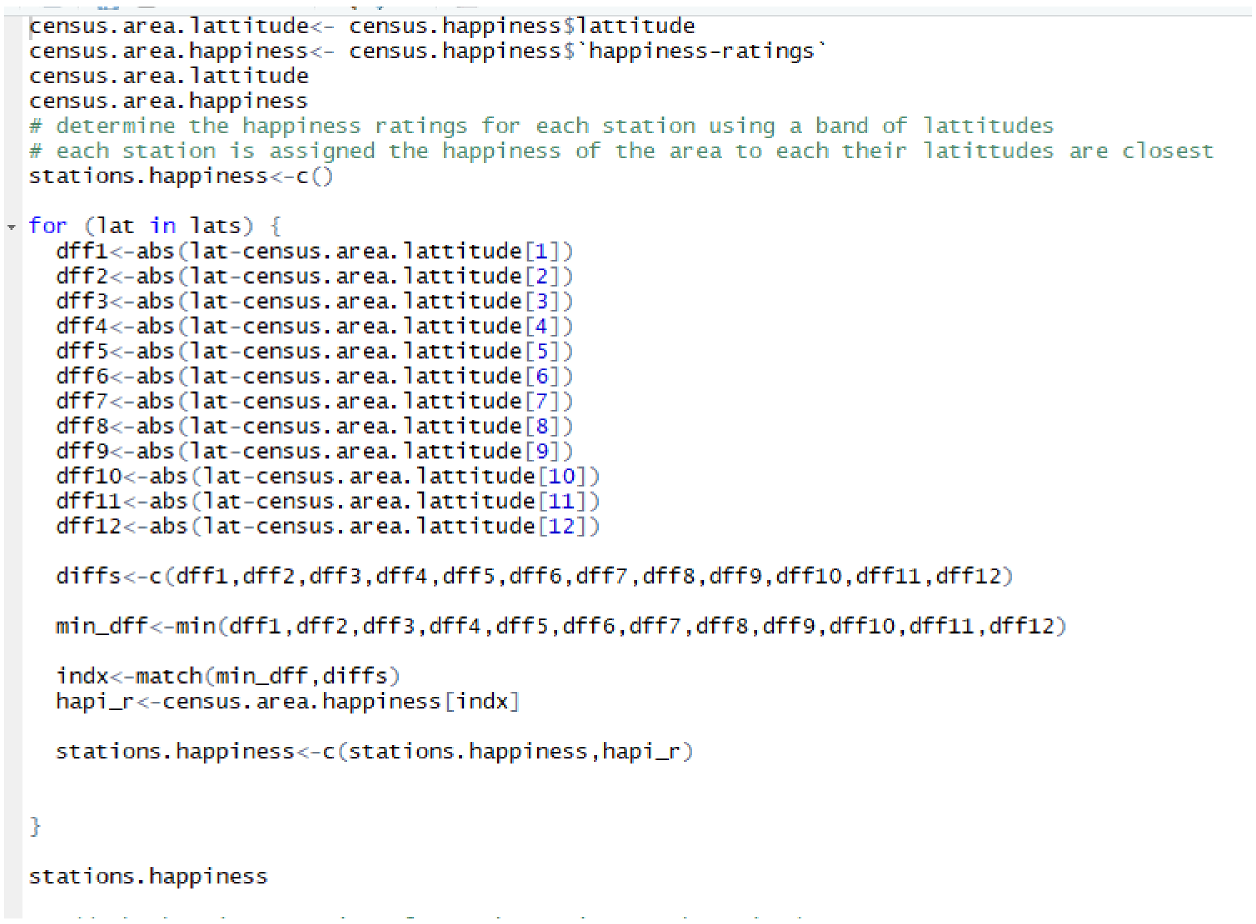
To find out if there is a correlation between weather and happiness levels, the weather data for each station must be merged with the happiness ratings for each station. Unfortunately, the happiness ratings from the census data do not exactly match the weather station locations. Therefore, the latitude of the census areas was used alongside the latitudes of the weather stations, to determine which census areas each station was closest to and assigned the appropriate happiness ratings to those stations.



```
> dataset_withLat
                     mean_maxT_degC mean_minT_degC mean_rain_mm latitude
aberporthdata             12.438009       7.227837     76.02313   52.139
armaghdata                12.943978       5.661667     69.12446   54.352
ballypatrickdata          11.738166       6.144379    110.94615   55.181
bradforddata              12.307061       5.737145     73.03922   53.813
braemardata               10.496756       2.746544     75.53004   57.006
cambornedata              13.424600       8.365600     89.95580   50.218
cambridgedata             14.262360       6.189185     46.10478   52.245
cardiffdata               14.625781       7.033789     97.11973   51.488
chivenordata              14.516024       7.856998     77.11318   51.089
cwmystwythdata            11.635081       4.925678    151.20362   52.358
dunstaffnagedata          12.373874       6.262523    140.53676   56.451
durhamdata                12.387463       4.854545     54.39816   54.768
eastbournedata            13.892527       8.323505     66.63723   50.762
eskdalemuirdata           10.924451       3.429545    134.88503   55.311
heathrowdata              14.903226       7.074309     50.50783   51.479
hurndata                  14.576447       5.893816     70.00026   50.779
lerwickdata                9.481361       5.051817     96.75154   60.139
leucharsdata              12.282368       4.963158     56.91947   56.377
lowestoftdata             13.046823       6.787124     50.16129   52.483
manstondata               13.732538       7.189154     49.20770   51.346
nairndata                 11.938454       4.893798     52.52099   57.593
newtonriggdata            12.138062       4.805337     78.52893   54.670
oxforddata                13.944743       6.213154     54.70688   51.761
paisleydata               12.710190       6.089946     99.58832   55.846
ringwaydata               13.070342       6.297529     67.89297   53.356
rossonwyedata             14.009972       6.221156     60.02591   51.911
shawburydata              13.442895       5.305789     55.94211   52.794
sheffielddata             12.881096       6.218966     66.59901   53.381
southamptondata           14.406239       6.757597     66.52819   50.898
stornowaydata             11.007631       5.356093    100.00313   58.214
suttonboningtondata       13.678792       5.883287     51.22430   52.833
tireedata                 11.639515       6.753961     98.93728   56.500
valleydata                13.001213       7.504011     71.09347   53.252
waddingtondata            13.231932       5.943750     50.56000   53.175
whitbydata                12.460294       6.189706     49.60368   54.481
wickairportdata           10.490284       5.010821     65.23973   58.454
yeoviltondata             14.512126       6.103144     60.58877   51.006
>
> #Load the census data with mean happiness ratings added
> census.happiness<-read.csv("dataset2/censusdata.txt", header = FALSE, sep=",",dec=".")
>
> # Get only the only the latitude and happiness ratings column
> census.happiness<-census.happiness[,c(3,5)]
> names(census.happiness)<-c("latitude","happiness-ratings")
> census.happiness
   latitude happiness-ratings
1      55.0              7.34
2      54.0              7.39
3      53.6              7.41
4      53.0              7.51
5      52.5              7.43
6      52.2              7.51
7      51.5              7.38
8      51.3              7.54
9      51.0              7.50
10     51.5              7.44
11     56.0              7.45
12     54.6              7.75
```

Figure 9: Weather dataset and extracted happiness data showing only census area latitude and happiness rating

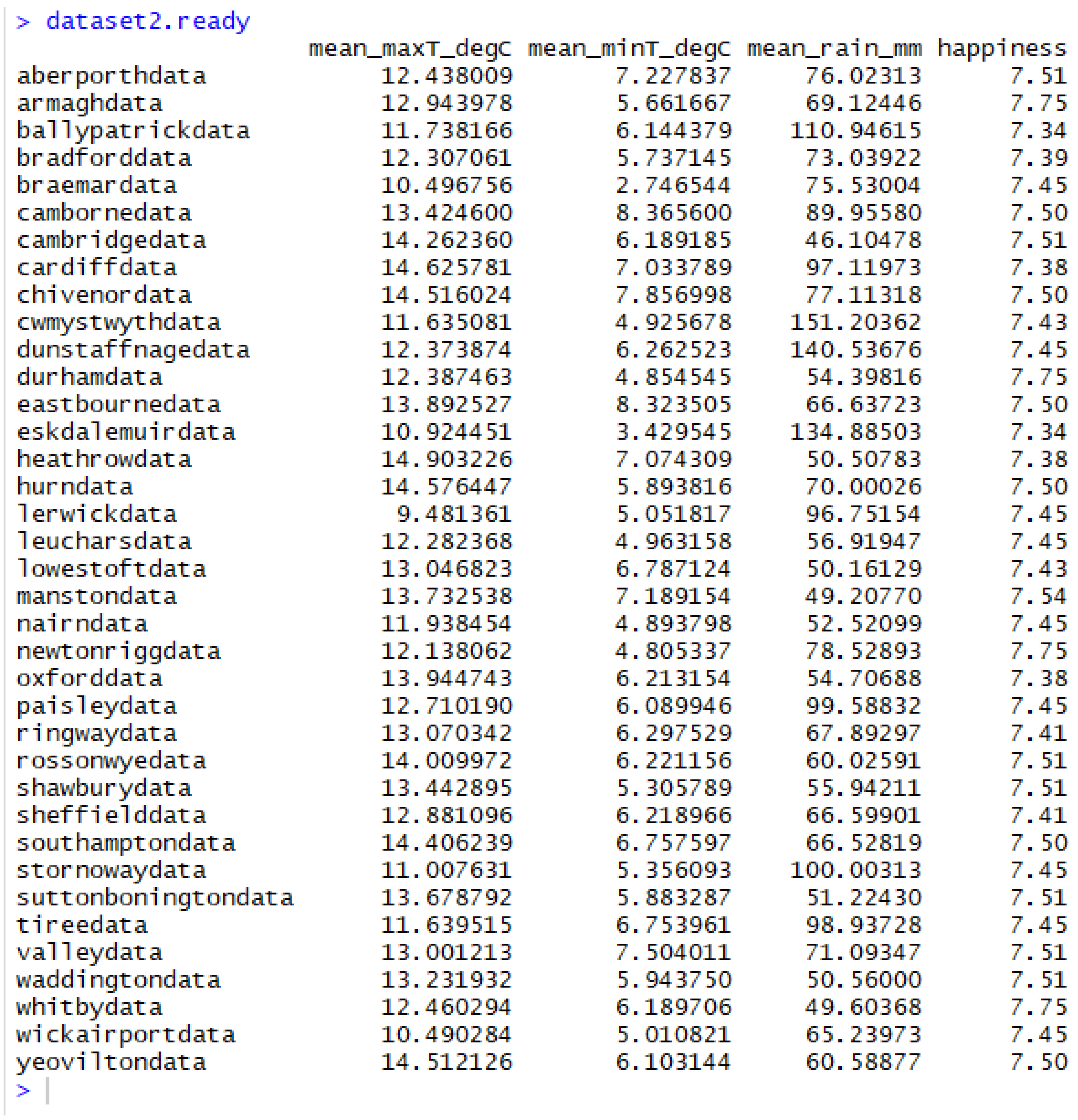
To assign the right happiness rating to each station, the following code was used. **To merge the datasets, it is assumed that the happiness ratings do not vary drastically from one year to another, and that climatic conditions are similar year after year.**



```
census.area.lattitude<- census.happiness$lattitude
census.area.happiness<- census.happiness$`happiness-ratings`
census.area.lattitude
census.area.happiness
# determine the happiness ratings for each station using a band of lattitudes
# each station is assigned the happiness of the area to each their lattitudes are closest
stations.happiness<-c()

for (lat in lats) {
  dff1<-abs(lat-census.area.lattitude[1])
  dff2<-abs(lat-census.area.lattitude[2])
  dff3<-abs(lat-census.area.lattitude[3])
  dff4<-abs(lat-census.area.lattitude[4])
  dff5<-abs(lat-census.area.lattitude[5])
  dff6<-abs(lat-census.area.lattitude[6])
  dff7<-abs(lat-census.area.lattitude[7])
  dff8<-abs(lat-census.area.lattitude[8])
  dff9<-abs(lat-census.area.lattitude[9])
  dff10<-abs(lat-census.area.lattitude[10])
  dff11<-abs(lat-census.area.lattitude[11])
  dff12<-abs(lat-census.area.lattitude[12])

  diffs<-c(dff1,dff2,dff3,dff4,dff5,dff6,dff7,dff8,dff9,dff10,dff11,dff12)

  min_dff<-min(dff1,dff2,dff3,dff4,dff5,dff6,dff7,dff8,dff9,dff10,dff11,dff12)

  indx<-match(min_dff,diffs)
  hapi_r<-census.area.happiness[indx]

  stations.happiness<-c(stations.happiness,hapi_r)


}

stations.happiness
```

The happiness ratings for each station was obtained and added to the main dataset to be used for correlation.

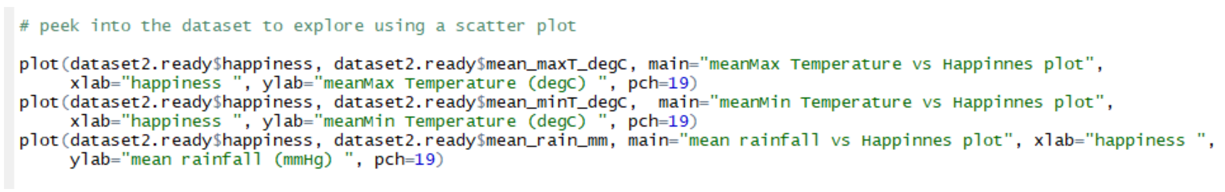
To find out if there is an association between a locations weather and how happy people in the location are, different types of coefficient of correlation can be used. However, it is important to explore the relationship between the variables by plotting them first. The weather instruments of interest here are temperature and rainfall. This project checks the association between mean minimum temperature and happiness, mean maximum temperature and happiness and mean rainfall and happiness separately. The ready dataset in figure 10 is used for the rest of this section.



```
> dataset2.ready
                    mean_maxT_degC mean_minT_degC mean_rain_mm happiness
aberporthdata            12.438009       7.227837     76.02313      7.51
armaghdata               12.943978       5.661667     69.12446      7.75
ballypatrickdata         11.738166       6.144379    110.94615      7.34
bradforddata             12.307061       5.737145     73.03922      7.39
braemardata              10.496756       2.746544     75.53004      7.45
camborneddata            13.424600       8.365600     89.95580      7.50
cambridgedata            14.262360       6.189185     46.10478      7.51
cardiffdata              14.625781       7.033789     97.11973      7.38
chivenordata             14.516024       7.856998     77.11318      7.50
cwmystwythdata           11.635081       4.925678    151.20362      7.43
dunstaffnagedata         12.373874       6.262523    140.53676      7.45
durhamdata               12.387463       4.854545     54.39816      7.75
eastbournedata           13.892527       8.323505     66.63723      7.50
eskdalemuirdata          10.924451       3.429545    134.88503      7.34
heathrowdata             14.903226       7.074309     50.50783      7.38
hurndata                 14.576447       5.893816     70.00026      7.50
lerwickdata               9.481361       5.051817     96.75154      7.45
leucharsdata             12.282368       4.963158     56.91947      7.45
lowestoftdata            13.046823       6.787124     50.16129      7.43
manstondata              13.732538       7.189154     49.20770      7.54
nairndata                11.938454       4.893798     52.52099      7.45
newtonriggdata           12.138062       4.805337     78.52893      7.75
oxforddata               13.944743       6.213154     54.70688      7.38
paisleydata              12.710190       6.089946     99.58832      7.45
ringwaydata              13.070342       6.297529     67.89297      7.41
rossonwyedata            14.009972       6.221156     60.02591      7.51
shawburydata             13.442895       5.305789     55.94211      7.51
sheffielddata            12.881096       6.218966     66.59901      7.41
southamptondata          14.406239       6.757597     66.52819      7.50
stornowaydata            11.007631       5.356093    100.00313      7.45
suttonboningtondata      13.678792       5.883287     51.22430      7.51
tireedata                11.639515       6.753961     98.93728      7.45
valleydata               13.001213       7.504011     71.09347      7.51
waddingtondata           13.231932       5.943750     50.56000      7.51
whitbydata               12.460294       6.189706     49.60368      7.75
wickairportdata          10.490284       5.010821     65.23973      7.45
yeoviltondata            14.512126       6.103144     60.58877      7.50
> 
```

Figure 10: Dataset weather and happiness data for each station, ready to be used for correlation

Plots of the various weather variables against the happiness ratings show interesting outputs.



```
# peek into the dataset to explore using a scatter plot

plot(dataset2.ready$happiness, dataset2.ready$mean_maxT_degC, main="meanMax Temperature vs Happinnes plot",
     xlab="happiness ", ylab="meanMax Temperature (degC) ", pch=19)
plot(dataset2.ready$happiness, dataset2.ready$mean_minT_degC,  main="meanMin Temperature vs Happinnes plot",
     xlab="happiness ", ylab="meanMin Temperature (degC) ", pch=19)
plot(dataset2.ready$happiness, dataset2.ready$mean_rain_mm, main="mean rainfall vs Happinnes plot", xlab="happiness ",
     ylab="mean rainfall (mmHg) ", pch=19)
```

To determine which coefficient of correlation to use (Pearson, spearman or kendall etc.), it is important to explore the data a little further and these plots help with that. From the three scatter plots below, it seems clear that there is no linear association between any of the variables and happiness, hence we cannot use Pearson coefficient of correlation here since it only does well in predicting relationships between two variables with linear association. But, before that conclusion, one last check for normality was done. Note that all happiness ratings are out of 10 (e.g. 7.5/10)

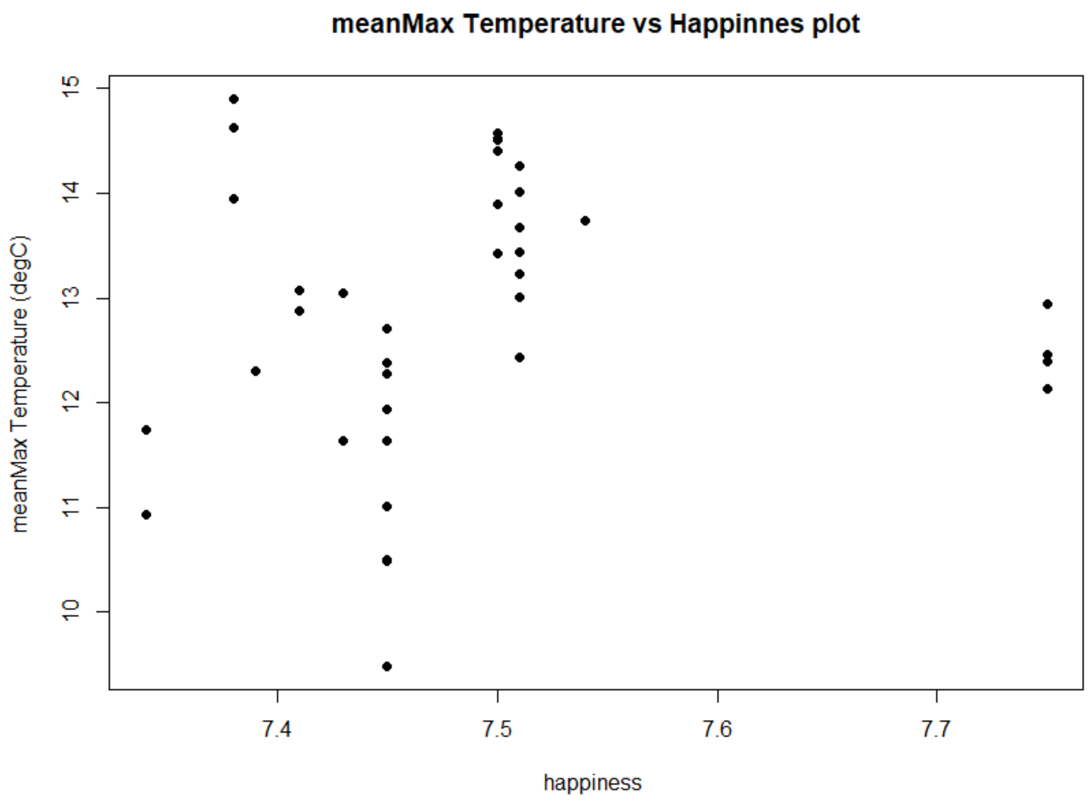


Figure 11: mean Max Temperature vs Happiness ratings

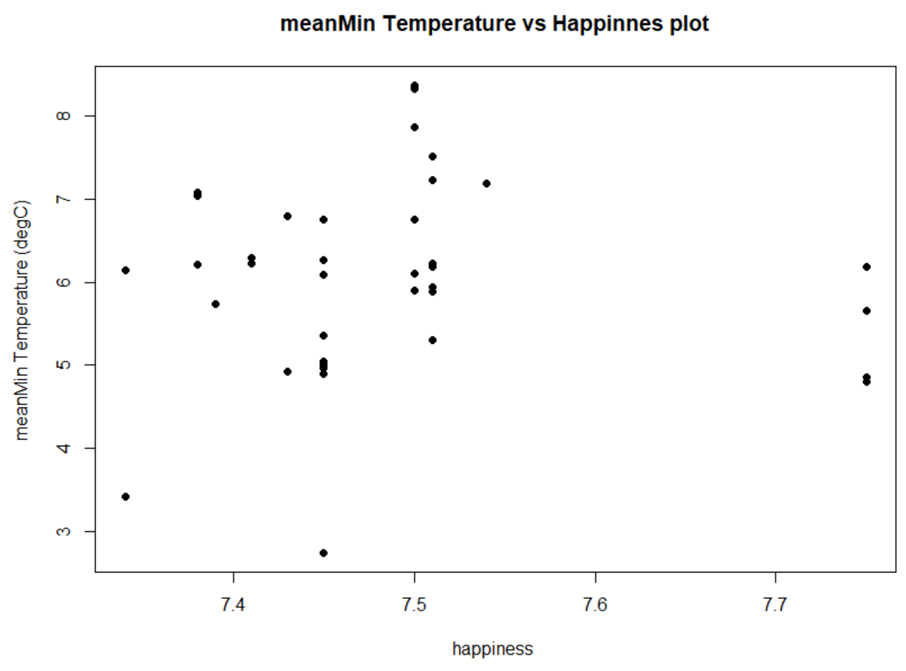


Figure 12:mean Min Temperature vs Happiness ratings

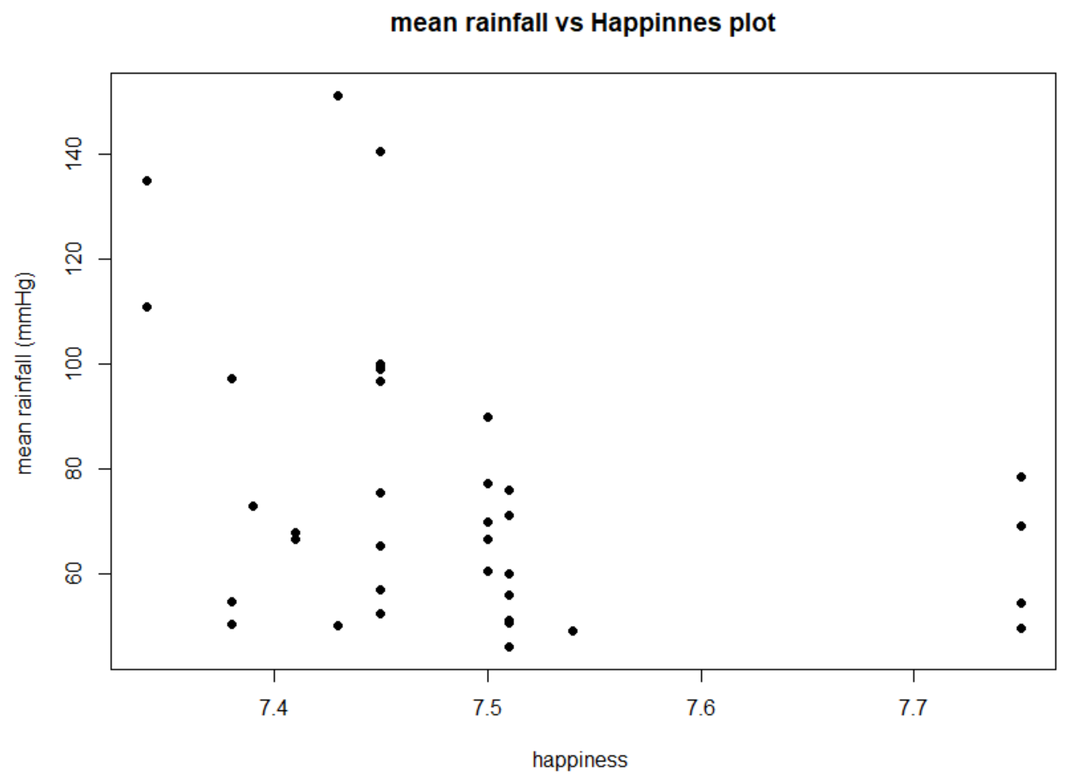
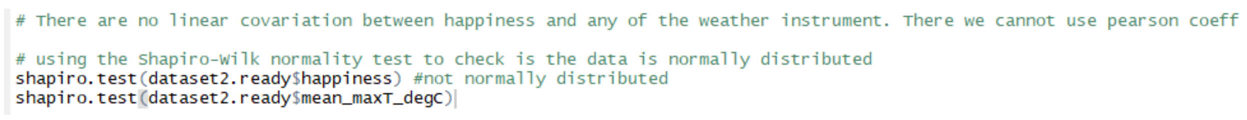


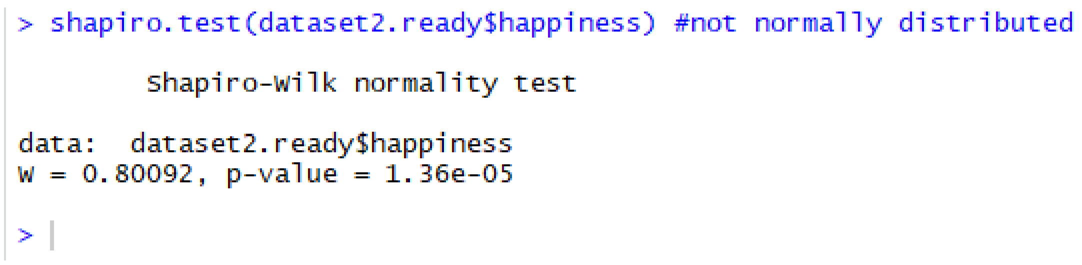
Figure 13: Mean rainfall and happiness plot

From the scatter plots, there is no linearity. To also check if the variables are normally distributed, the Shapiro-Wilk normality test was used.



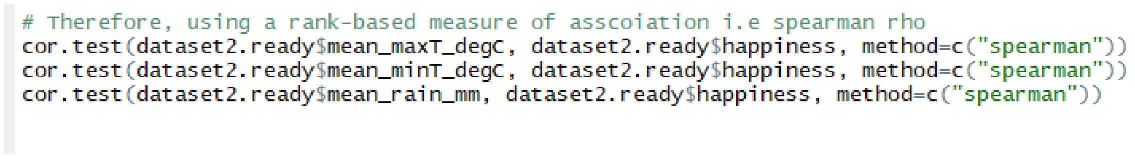
```
# There are no linear covariation between happiness and any of the weather instrument. There we cannot use pearson coeff

# using the Shapiro-Wilk normality test to check is the data is normally distributed
shapiro.test(dataset2.ready$happiness) #not normally distributed
shapiro.test(dataset2.ready$mean_maxT_degC)
```

The results of the normality tests showed that the happiness data is not normally distributed. Hence, Pearson coefficient is completely out of the options since it gives a p-value less than 0.05 (limit for statistical significance).



```
> shapiro.test(dataset2.ready$happiness) #not normally distributed

        Shapiro-Wilk normality test

data:  dataset2.ready$happiness
W = 0.80092, p-value = 1.36e-05

> 
```

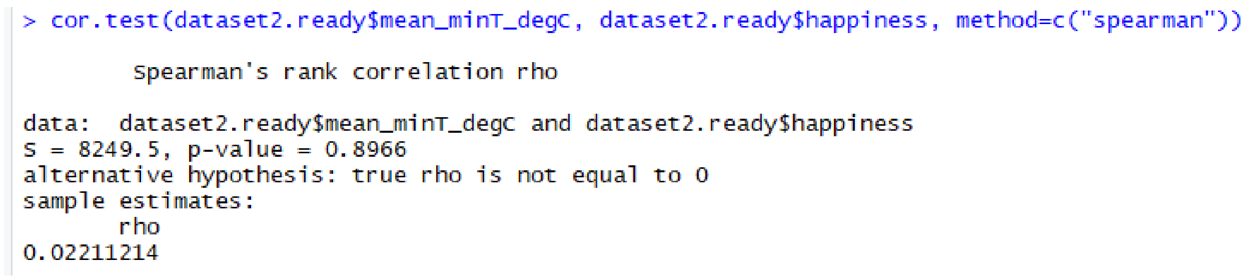
Because both the happiness and weather data are not normally distributed and there is no linearity between them, a rank-based coefficient of correlation, the Spearman coefficient of correlation (rho) was used with the R built-in function cor.test().



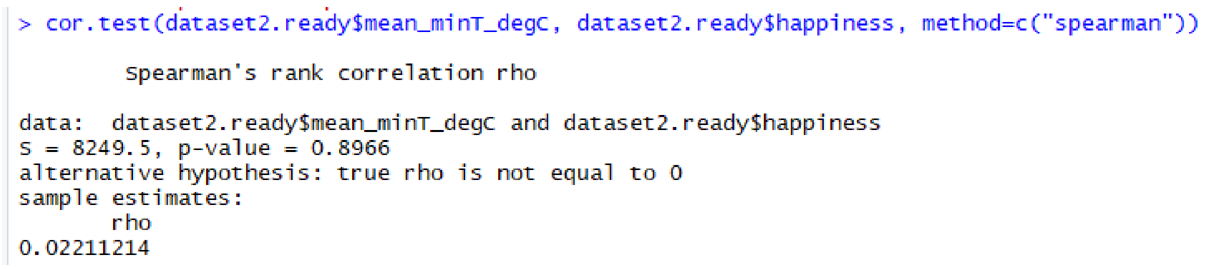
```
# Therefore, using a rank-based measure of asscoiation i.e spearman rho
cor.test(dataset2.ready$mean_maxT_degC, dataset2.ready$happiness, method=c("spearman"))
cor.test(dataset2.ready$mean_minT_degC, dataset2.ready$happiness, method=c("spearman"))
cor.test(dataset2.ready$mean_rain_mm, dataset2.ready$happiness, method=c("spearman"))
```

Result

The rho shows a loose association between weather and happiness. The rho values for temperature and happiness is close to zero, which indicates that there may be a random positive relationship or just no relationship at all between weather with respect to temperatures and how happy people feel.

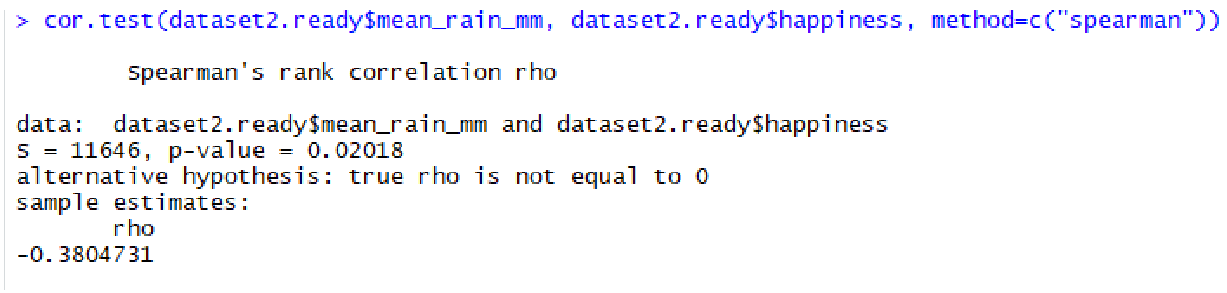


```
> cor.test(dataset2.ready$mean_minT_degC, dataset2.ready$happiness, method=c("spearman"))

        Spearman's rank correlation rho

data:  dataset2.ready$mean_minT_degC and dataset2.ready$happiness
S = 8249.5, p-value = 0.8966
alternative hypothesis: true rho is not equal to 0
sample estimates:
       rho 
0.02211214 
```



```
> cor.test(dataset2.ready$mean_minT_degC, dataset2.ready$happiness, method=c("spearman"))

	Spearman's rank correlation rho

data:  dataset2.ready$mean_minT_degC and dataset2.ready$happiness
S = 8249.5, p-value = 0.8966
alternative hypothesis: true rho is not equal to 0
sample estimates:
       rho 
0.02211214 
```

On the other hand, people tend to say they feel happier when it does not rain too much. This association is not as loose as that between temperature and happiness, but it is negative and seems to make sense that when it rains too much people are not too happy. Again, the rho value is not encouraging to say with firmness that the weather with respect to rainfall affects how happy people feel.



```
> cor.test(dataset2.ready$mean_rain_mm, dataset2.ready$happiness, method=c("spearman"))

	Spearman's rank correlation rho

data:  dataset2.ready$mean_rain_mm and dataset2.ready$happiness
S = 11646, p-value = 0.02018
alternative hypothesis: true rho is not equal to 0
sample estimates:
       rho 
-0.3804731 
```

Assignment part 4b: Automating the project using R

The project was automated using a single R script that makes it easy to run the entire project and step through to reproduce steps and results easily and efficiently.

In the attached project folder, there are:

- Two folders for “dataset1” and “dataset2” which contain files for weather data and intermediary happiness data respectively
- An R file called “ass3_all_parts” which contains R script that automates this entire project

Instructions

1- Open RStudio or any other IDE that supports the R language

2- Change the working directory to point to the folder

3- Load the R file “ass3_all_parts”. If for some reasons the file opens in RStudio, but it looks empty, go to “File->Reopen with encoding” and select UTF-8

The script is divided into various sections. The first section shows the libraries you might need to install in other to run the project. Install those that you do not have yet. Note that these libraries may depend on other libraries that you might not have installed. Move to the next section in Part 1 and run all the functions until you get to the section that says “FUNCTION CALLS”

Once you call the function k_means_cluster() and pass it the directory to dataset1, it will run all the functions to prepare and clean the data and do the k-means clustering for question one. Some results are printed in the console, so please check. This is all for Part 1 of the assignment.

Part 2 and 3 of the assignment allows you to run the code anyhow you want – whether line by line or in one batch.

Please refer to the comments and guides on the script file to follow the code.

Conclusion

In this project, all the above objectives were met. The project acquired data from the two data sources, cleaned and prepared them, then successfully used k-means clustering to find 3 clusters of weather stations with similar weather. A KNN classification model was implemented and it got a 60% accuracy for classifying different weather stations into their various regions (northern third, central third or southern third) given only weather data. The project also used a Spearman coefficient of correlation to establish that there is a very loose (or random) relationship between weather and how happy people feel. Using only temperature data, the correlation is positive, but with rainfall data, there is a negative correlation with happiness.

Finally, this project was automated to allow anyone to be able to easily run and reproduce the steps and results obtained in this report. All that is needed is a machine with R and RStudio, and the zip file containing this project’s scripts and datasets.