Study Area Mapping & Regression Machine Learning

Zarrin Sovha Khan

```r
library(sf)
```

```
## Linking to GEOS 3.12.2, GDAL 3.9.3, PROJ 9.4.1; sf_use_s2() is TRUE
```

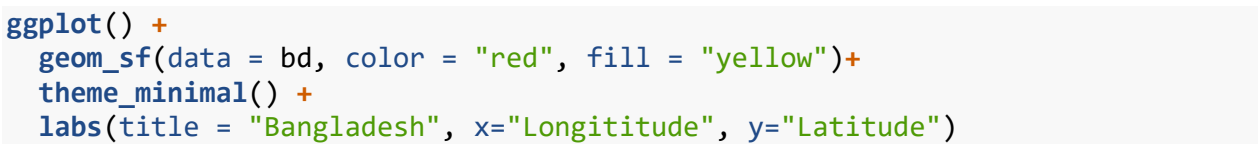
```r
library(installr)
```

```
##
## Welcome to installr version 0.23.4
##
## More information is available on the installr project website:
## https://github.com/talgalili/installr/
##
## Contact: <tal.galili@gmail.com>
## Suggestions and bug-reports can be submitted at:
https://github.com/talgalili/installr/issues
##
##          To suppress this message use:
##          suppressPackageStartupMessages(library(installr))
```

```r
library(readxl)
Lentil_data <- read_excel("D:/Data Analysis/Map/Lentil data.xlsx")

bd <-st_read("D:/Data Analysis/Map/BGD_adm/BGD_adm3.shp")
```

```
## Reading layer `BGD_adm3' from data source
##   `D:\Data Analysis\Map\BGD_adm\BGD_adm3.shp' using driver `ESRI
Shapefile'
## Simple feature collection with 463 features and 13 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: 88.01057 ymin: 20.74111 xmax: 92.67366 ymax: 26.6344
## Geodetic CRS:  WGS 84
```

```r
library(ggplot2)

ggplot() +
  geom_sf(data = bd)
```



```r
ggplot() +
  geom_sf(data = bd, color = "red", fill = "yellow")+
  theme_minimal() +
  labs(title = "Bangladesh", x="Longititude", y="Latitude")
```

```r
sites <- st_as_sf(Lentil_data, coords = c("Lon", "Lat"))

st_crs(sites)

## Coordinate Reference System: NA

st_crs(sites) <-st_crs(bd)

st_crs(sites)

## Coordinate Reference System:
##   User input: WGS 84
##   wkt:
## GEOGCRS["WGS 84",
##     DATUM["World Geodetic System 1984",
##         ELLIPSOID["WGS 84",6378137,298.257223563,
##             LENGTHUNIT["metre",1]]],
##     PRIMEM["Greenwich",0,
##         ANGLEUNIT["degree",0.0174532925199433]],
##     CS[ellipsoidal,2],
##         AXIS["latitude",north,
##             ORDER[1],
##             ANGLEUNIT["degree",0.0174532925199433]],
##         AXIS["longitude",east,
##             ORDER[2],
##             ANGLEUNIT["degree",0.0174532925199433]],
##     ID["EPSG",4326]]

ggplot() +
  geom_sf(data = bd, color = "red", fill = "yellow")+
  geom_sf(data = sites, color="green") +
  theme_minimal() +
  labs(title = "Bangladesh", x="Longititude", y="Latitude")
```

```r
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

head(bd, 2)
```

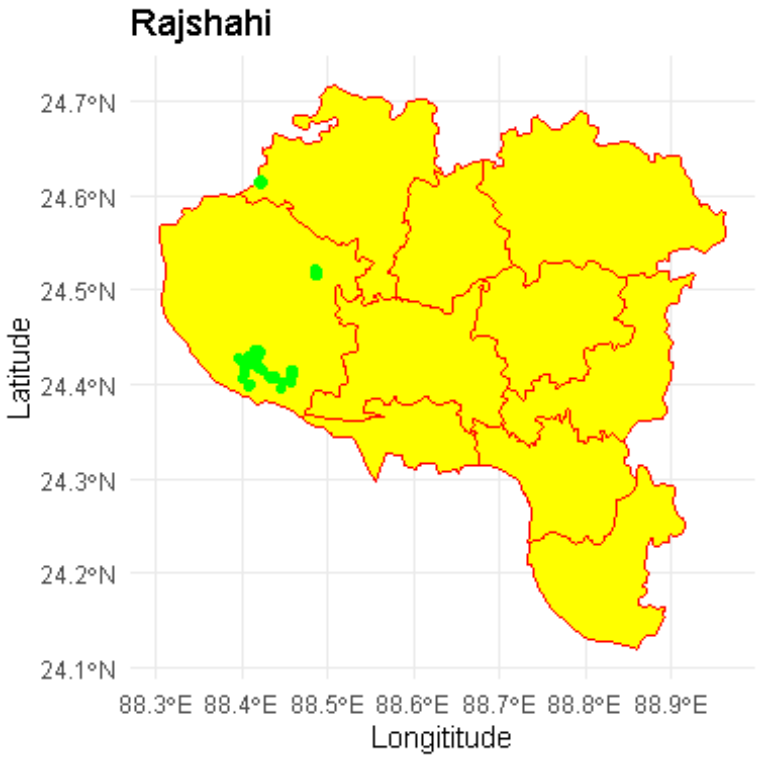
```
## Simple feature collection with 2 features and 13 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: 90.05692 ymin: 22.73647 xmax: 90.38506 ymax: 23.03652
## Geodetic CRS:  WGS 84
##   ID_0 ISO     NAME_0 ID_1  NAME_1 ID_2  NAME_2 ID_3     NAME_3
## 1   20 BGD Bangladesh    1 Barisal    1 Barisal    1 Agailjhara
## 2   20 BGD Bangladesh    1 Barisal    1 Barisal    2   Babuganj
##             TYPE_3    ENGTYPE_3 NL_NAME_3 VARNAME_3
## 1 Upazila|Thana|Po Sub-district      <NA>      <NA>
## 2 Upazila|Thana|Po Sub-district      <NA>      <NA>
##                           geometry
## 1 MULTIPOLYGON (((90.13228 23...
## 2 MULTIPOLYGON (((90.27067 22...
```

```
bd %>% filter(NAME_2 =="Rajshahi")
```

```
## Simple feature collection with 10 features and 13 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: 88.30386 ymin: 24.11916 xmax: 88.96407 ymax: 24.71719
## Geodetic CRS:  WGS 84
##    ID_0 ISO     NAME_0 ID_1   NAME_1 ID_2   NAME_2 ID_3   NAME_3
## 1    20 BGD Bangladesh    5 Rajshahi   52 Rajshahi  352    Bagha
## 2    20 BGD Bangladesh    5 Rajshahi   52 Rajshahi  353  Bagmara
## 3    20 BGD Bangladesh    5 Rajshahi   52 Rajshahi  354   Boalia
## 4    20 BGD Bangladesh    5 Rajshahi   52 Rajshahi  355 Charghat
## 5    20 BGD Bangladesh    5 Rajshahi   52 Rajshahi  356 Durgapur
## 6    20 BGD Bangladesh    5 Rajshahi   52 Rajshahi  357 Godagari
## 7    20 BGD Bangladesh    5 Rajshahi   52 Rajshahi  358 Mohanpur
## 8    20 BGD Bangladesh    5 Rajshahi   52 Rajshahi  359     Paba
## 9    20 BGD Bangladesh    5 Rajshahi   52 Rajshahi  360   Puthia
## 10   20 BGD Bangladesh    5 Rajshahi   52 Rajshahi  361   Tanore
##              TYPE_3    ENGTYPE_3 NL_NAME_3 VARNAME_3
## 1  Upazila|Thana|Po Sub-district      <NA>      <NA>
## 2  Upazila|Thana|Po Sub-district      <NA>      <NA>
## 3  Upazila|Thana|Po Sub-district      <NA>      <NA>
## 4  Upazila|Thana|Po Sub-district      <NA>      <NA>
## 5  Upazila|Thana|Po Sub-district      <NA>  Rajshahi
## 6  Upazila|Thana|Po Sub-district      <NA>      <NA>
## 7  Upazila|Thana|Po Sub-district      <NA>      <NA>
## 8  Upazila|Thana|Po Sub-district      <NA>      <NA>
## 9  Upazila|Thana|Po Sub-district      <NA>      <NA>
## 10 Upazila|Thana|Po Sub-district      <NA>      <NA>
##                           geometry
## 1  MULTIPOLYGON (((88.8597 24....
## 2  MULTIPOLYGON (((88.8864 24....
## 3  MULTIPOLYGON (((88.67597 24...
## 4  MULTIPOLYGON (((88.72646 24...
## 5  MULTIPOLYGON (((88.70962 24...
```

```
## 6  MULTIPOLYGON (((88.47128 24...
## 7  MULTIPOLYGON (((88.58035 24...
## 8  MULTIPOLYGON (((88.70962 24...
## 9  MULTIPOLYGON (((88.8511 24....
## 10 MULTIPOLYGON (((88.39628 24...
```

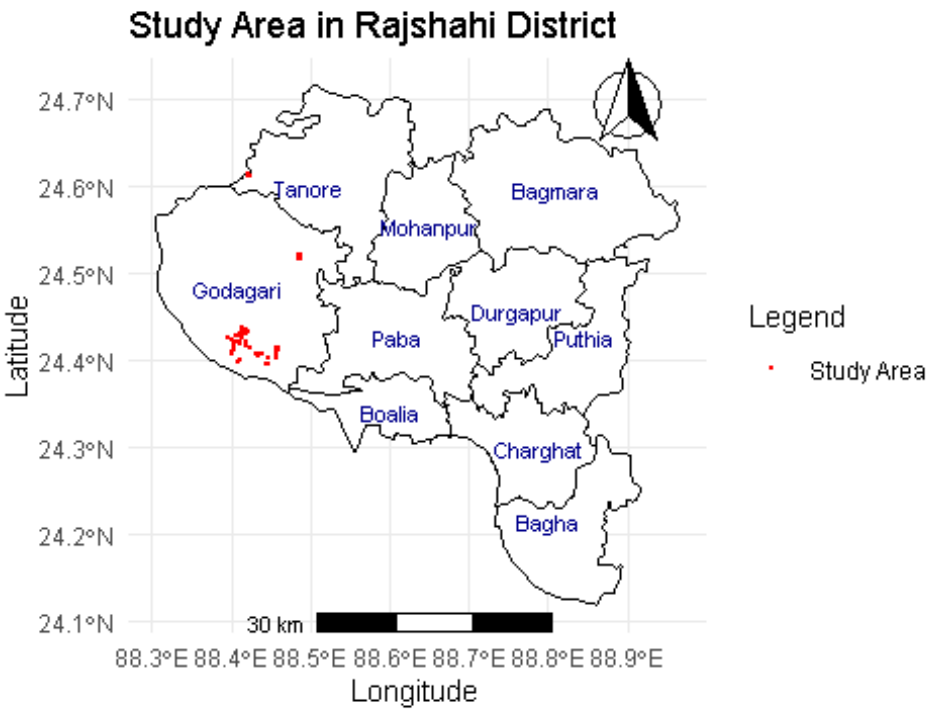
```r
rj <-bd %>% filter(NAME_2 =="Rajshahi")

ggplot() +
  geom_sf(data = rj, color = "red", fill = "yellow")+
  geom_sf(data = sites, color="green") +
  theme_minimal() +
  labs(title = "Rajshahi", x="Longititude", y="Latitude")
```




```r
library(ggspatial)

ggplot() +
 # Plot Rajshahi district
 geom_sf(data = rj, fill = "white", color = "black") +
 # Add sub-district labels
 geom_sf_text(data = rj, aes(label = NAME_3), size = 3, color = "darkblue") +
 # Add study area point with legend
 geom_sf(data = sites, aes(color = "Study Area"), size = 0.5) +
 # Add the north arrow
 annotation_north_arrow(
 location = "tr", # Top-right corner
 which_north = "true", # True north
```

```
 style = north_arrow_fancy_orienteering(),
 pad_y = unit(-0.3, "cm") # Vertical padding (distance from the bottom edge)
 ) +
 # Add a scale bar
 annotation_scale(
 location = "br", # Bottom-left corner
 width_hint = 0.5, # Width of the scale bar
 pad_x = unit(2, "cm"), # Horizontal padding
 pad_y = unit(0, "cm") # Vertical padding (distance from the bottom edge)
 ) +
 # Enhance visualization
 theme_minimal() +
 # Add legend title and labels
 scale_color_manual(
 name = "Legend", # Legend title
 values = c("Study Area" = "red") # Define colors
 ) +
 labs(
 title = "Study Area in Rajshahi District",
 x = "Longitude",
 y = "Latitude")
```

```
## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may
not
## give correct results for longitude/latitude data
```



Machine Learning

ML_Regression

```
library(caret)
```

```
## Loading required package: lattice
```

```
library(randomForest)
```

```
## randomForest 4.7-1.2
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(e1071)
library(xgboost)
```

```
##
## Attaching package: 'xgboost'
```

```
## The following object is masked from 'package:dplyr':
##
##     slice
```

```
library(nnet)
library(Metrics)
```

```
##
## Attaching package: 'Metrics'
```

```
## The following objects are masked from 'package:caret':
##
##     precision, recall
```

```
library(ggplot2)
```

```
df <-Lentil_data[sample(nrow(Lentil_data)), ]
```

```
createDataPartition(df$yield, p=0.8, list = F)
```

```
##       Resample1
##  [1,]         1
```

```
##  [2,]         2
##  [3,]         3
##  [4,]         4
##  [5,]         6
##  [6,]         7
##  [7,]         8
##  [8,]         9
##  [9,]        10
## [10,]        11
## [11,]        12
## [12,]        13
## [13,]        15
## [14,]        16
## [15,]        17
## [16,]        18
## [17,]        19
## [18,]        20
## [19,]        21
## [20,]        22
## [21,]        23
## [22,]        24
## [23,]        25
## [24,]        26
## [25,]        27
## [26,]        29
## [27,]        30
## [28,]        31
## [29,]        32
## [30,]        33
## [31,]        35
## [32,]        36
## [33,]        37
## [34,]        38
## [35,]        39
## [36,]        40
## [37,]        41
## [38,]        42
## [39,]        43
## [40,]        44
## [41,]        46
## [42,]        47
## [43,]        48
## [44,]        49
## [45,]        50
## [46,]        53
## [47,]        54
## [48,]        55
## [49,]        56
## [50,]        57
## [51,]        58
```

```
## [52,]        59
## [53,]        60
## [54,]        61
## [55,]        62
## [56,]        63
## [57,]        65
## [58,]        67
## [59,]        68
## [60,]        72
## [61,]        73
## [62,]        74
## [63,]        75
## [64,]        76
## [65,]        78
## [66,]        79
## [67,]        80
## [68,]        81
## [69,]        82
## [70,]        85
## [71,]        86
## [72,]        87
## [73,]        88
## [74,]        89
## [75,]        90
## [76,]        91
## [77,]        93
## [78,]        94
## [79,]        96
## [80,]        97
## [81,]        98
## [82,]        99
## [83,]       100
## [84,]       101
## [85,]       102
## [86,]       103
## [87,]       104
## [88,]       106
## [89,]       107
## [90,]       108
## [91,]       109
## [92,]       111
## [93,]       113
## [94,]       114
## [95,]       115
## [96,]       117
```

```
trainIndex <-createDataPartition(df$yield, p=0.8, list = F)

df[trainIndex, ]
```

```
## # A tibble: 96 × 6
##       sl   Lat   Lon yield  NDVI   EVI
##    <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1    81  24.4  88.4  0.54 0.148 0.113
##  2    44  24.4  88.5  0.81 0.191 0.148
##  3   113  24.6  88.4  1.8  0.400 0.311
##  4    87  24.4  88.4  0.77 0.190 0.146
##  5    73  24.4  88.4  1.13 0.293 0.224
##  6    18  24.4  88.4  1.26 0.308 0.235
##  7   114  24.6  88.4  2.07 0.442 0.365
##  8    16  24.4  88.4  0.97 0.254 0.188
##  9    64  24.4  88.4  1.44 0.339 0.258
## 10    34  24.4  88.5  1.62 0.347 0.263
## # ⓘ 86 more rows
```

```
df[trainIndex, 4:6]
```

```
## # A tibble: 96 × 3
##    yield  NDVI   EVI
##    <dbl> <dbl> <dbl>
##  1  0.54 0.148 0.113
##  2  0.81 0.191 0.148
##  3  1.8  0.400 0.311
##  4  0.77 0.190 0.146
##  5  1.13 0.293 0.224
##  6  1.26 0.308 0.235
##  7  2.07 0.442 0.365
##  8  0.97 0.254 0.188
##  9  1.44 0.339 0.258
## 10  1.62 0.347 0.263
## # ⓘ 86 more rows
```

```
trainData <- df[trainIndex, 4:6]
testData <- df[-trainIndex, 4:6]

evalute_model <- function(actual, predicted) {
  r2 <- cor(actual, predicted)^2
  rmse <- RMSE(actual, predicted)
  return(data.frame(R2 = r2, RMSE = rmse))
}
```

Linear Regression

```
lm(yield~ NDVI + EVI, data = trainData)

##
## Call:
## lm(formula = yield ~ NDVI + EVI, data = trainData)
##
## Coefficients:
```

```
## (Intercept)         NDVI          EVI
##     -0.1736      -0.1733       6.4888
```

```
lm_model <- lm(yield~ NDVI + EVI, data = trainData)
lm_pred <- predict(lm_model, newdata = testData)
evalute_model(testData$yield, lm_pred)
```

```
##          R2       RMSE
## 1 0.9786844 0.06523576
```

```
lm_matrics <- evalute_model(testData$yield, lm_pred)
lm_matrics
```

```
##          R2       RMSE
## 1 0.9786844 0.06523576
```

Random Forrest

```
rf_model <- randomForest(yield ~ NDVI + EVI, data = trainData, ntree = 100)
rf_pred <- predict(rf_model, newdata = testData)
rf_metrics <- evalute_model(testData$yield, rf_pred)
rf_metrics
```

```
##          R2       RMSE
## 1 0.9842794 0.05589708
```

Support Vector Machine (SVM)

```
svm_model <- svm(yield ~ NDVI + EVI, data = trainData)
svm_pred <- predict(svm_model, newdata = testData)
svm_metrics <- evalute_model(testData$yield, svm_pred)
svm_metrics
```

```
##         R2       RMSE
## 1 0.957307 0.09076853
```

Artificial neural network (ANN)

```
ann_model <- nnet(yield ~ NDVI + EVI, data = trainData, size = 5,
                  linout = TRUE, trace = FALSE)
ann_pred<-predict(ann_model,newdata = testData)
ann_matrics <- evalute_model(testData$yield, ann_pred)
ann_matrics
```

```
##         R2       RMSE
## 1 0.980345 0.06221138
```

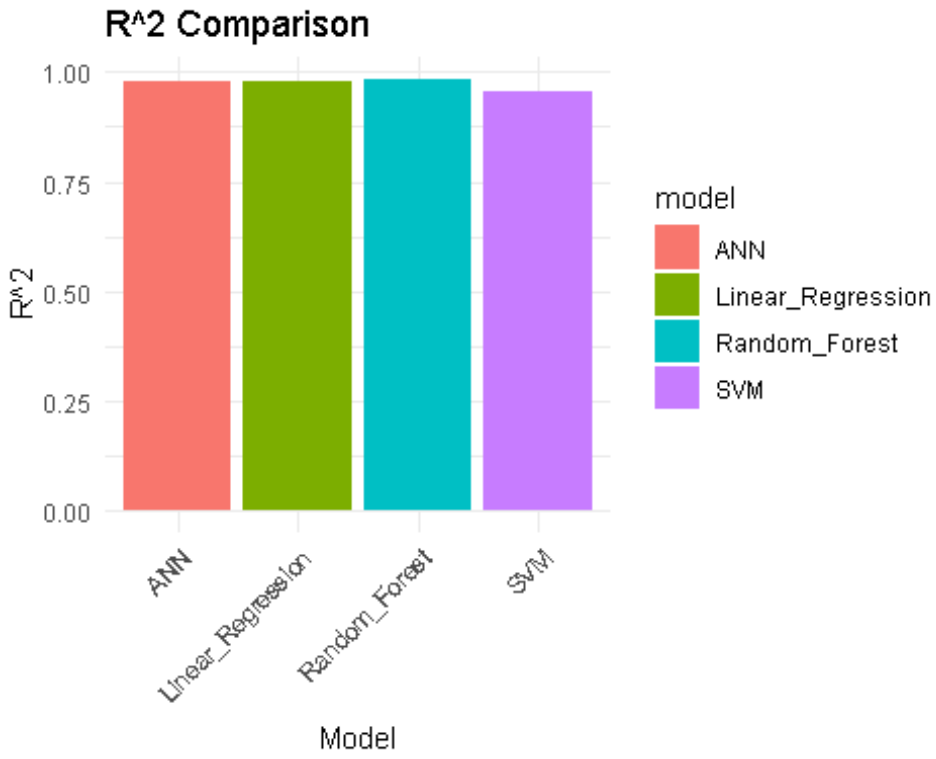
Comparison of model

```
model_comparison<-rbind(
  Linear_Regression = lm_matrics,
  Random_Forest = rf_metrics,
  SVM=svm_metrics,
```

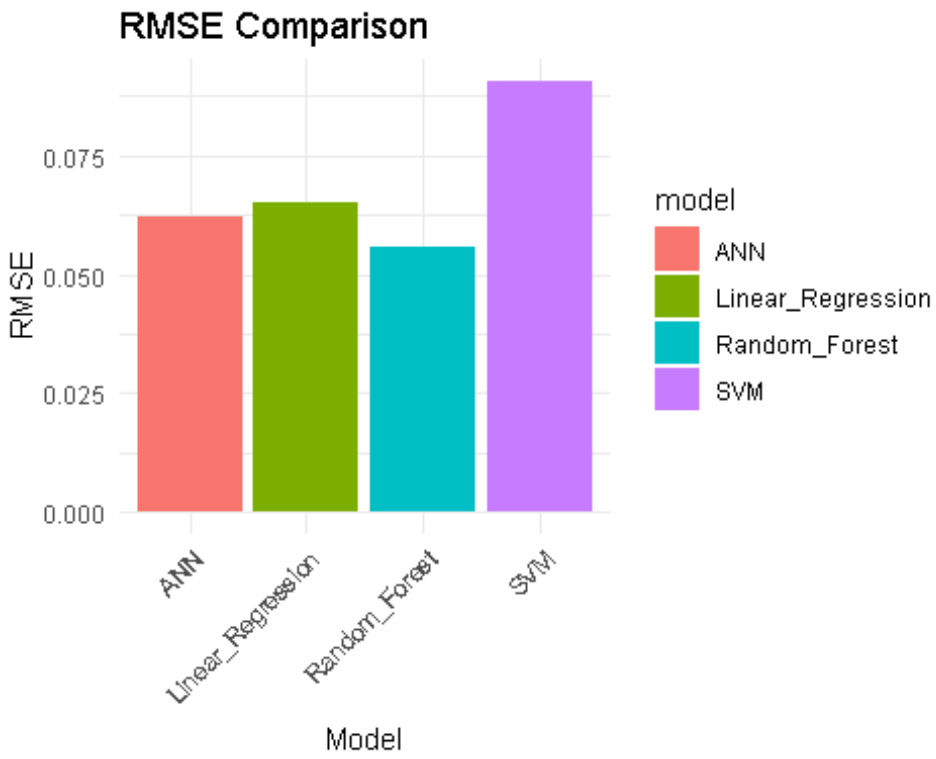
```
  ANN=ann_matrics
)

print(model_comparison)
```

```
##                            R2       RMSE
## Linear_Regression 0.9786844 0.06523576
## Random_Forest     0.9842794 0.05589708
## SVM               0.9573070 0.09076853
## ANN               0.9803450 0.06221138
```

```
model_comparison$model <- rownames(model_comparison)
ggplot(model_comparison, aes(x = model, y = R2, fill = model)) +
 geom_bar(stat = "identity") +
 theme_minimal() +
 labs(title = "R^2 Comparison", y = "R^2", x = "Model") +
 theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```
ggplot(model_comparison, aes(x = model, y = RMSE, fill = model)) +
 geom_bar(stat = "identity") +
 theme_minimal() +
 labs(title = "RMSE Comparison", y = "RMSE", x = "Model") +
 theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



ML Classification (supervised)

```
library(readxl)
Data <- read_excel("D:/Data Analysis/Map/WinnipegDataset.xlsx")

library(rpart)

Data$label <- factor(Data$label)

data <- Data[sample(nrow(Data)), ]

train_index <- createDataPartition(data$label, p = 0.8, list = FALSE)
train_data <- data[train_index, ]
test_data <- data[-train_index, ]
```

Decision Tree

```
tree_model <- rpart(label ~ ., data = train_data, method = "class")
tree_pred <- predict(tree_model, test_data, type = "class")
tree_cm <- confusionMatrix(tree_pred, test_data$label)
tree_cm

## Confusion Matrix and Statistics
##
##            Reference
## Prediction  Broadleaf Canola Corn Oat Pea Soy Wheat
##   Broadleaf       140      0    2   4   0   2    10
##   Canola            3    184    0   0   2   0     0
```

```
##   Corn             12      0  175   2   0   0     5
##   Oat              41     11   17 178   6  10    99
##   Pea               0      5    0   1 189   0     1
##   Soy               2      0    6   1   3 188    12
##   Wheat             2      0    0  14   0   0    73
##
## Overall Statistics
##
##                Accuracy : 0.805
##                  95% CI : (0.7833, 0.8255)
##     No Information Rate : 0.1429
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7725
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Broadleaf Class: Canola Class: Corn Class: Oat
## Sensitivity                    0.7000        0.9200      0.8750     0.8900
## Specificity                    0.9850        0.9958      0.9842     0.8467
## Pos Pred Value                 0.8861        0.9735      0.9021     0.4917
## Neg Pred Value                 0.9517        0.9868      0.9793     0.9788
## Prevalence                     0.1429        0.1429      0.1429     0.1429
## Detection Rate                 0.1000        0.1314      0.1250     0.1271
## Detection Prevalence           0.1129        0.1350      0.1386     0.2586
## Balanced Accuracy              0.8425        0.9579      0.9296     0.8683
##                      Class: Pea Class: Soy Class: Wheat
## Sensitivity              0.9450     0.9400      0.36500
## Specificity              0.9942     0.9800      0.98667
## Pos Pred Value           0.9643     0.8868      0.82022
## Neg Pred Value           0.9909     0.9899      0.90313
## Prevalence               0.1429     0.1429      0.14286
## Detection Rate           0.1350     0.1343      0.05214
## Detection Prevalence     0.1400     0.1514      0.06357
## Balanced Accuracy        0.9696     0.9600      0.67583
```

Random Forest

```
rf_model <- randomForest(label ~ ., data = train_data, importance = TRUE)
rf_pred <- predict(rf_model, test_data)
rf_cm <- confusionMatrix(rf_pred, test_data$label)
rf_cm
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  Broadleaf Canola Corn Oat Pea Soy Wheat
##   Broadleaf       169      0    0  11   1   0     2
```

```
##   Canola            5    196    0   1   1   0     1
##   Corn              7      0  195   1   0   0     6
##   Oat              13      1    0 163   1   0    28
##   Pea               1      3    0   0 195   0     0
##   Soy               1      0    1   0   2 194     7
##   Wheat             4      0    4  24   0   6   156
##
## Overall Statistics
##
##                Accuracy : 0.9057
##                  95% CI : (0.8892, 0.9205)
##     No Information Rate : 0.1429
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.89
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Broadleaf Class: Canola Class: Corn Class: Oat
## Sensitivity                    0.8450        0.9800      0.9750     0.8150
## Specificity                    0.9883        0.9933      0.9883     0.9642
## Pos Pred Value                 0.9235        0.9608      0.9330     0.7913
## Neg Pred Value                 0.9745        0.9967      0.9958     0.9690
## Prevalence                     0.1429        0.1429      0.1429     0.1429
## Detection Rate                 0.1207        0.1400      0.1393     0.1164
## Detection Prevalence           0.1307        0.1457      0.1493     0.1471
## Balanced Accuracy              0.9167        0.9867      0.9817     0.8896
##                      Class: Pea Class: Soy Class: Wheat
## Sensitivity              0.9750     0.9700       0.7800
## Specificity              0.9967     0.9908       0.9683
## Pos Pred Value           0.9799     0.9463       0.8041
## Neg Pred Value           0.9958     0.9950       0.9635
## Prevalence               0.1429     0.1429       0.1429
## Detection Rate           0.1393     0.1386       0.1114
## Detection Prevalence     0.1421     0.1464       0.1386
## Balanced Accuracy        0.9858     0.9804       0.8742
```

Train SVM

```
svm_model <- svm(label ~ ., data = train_data, kernel = "radial", probability
= TRUE)
```

SVM Predictions

```
svm_pred <- predict(svm_model, test_data)
svm_cm <- confusionMatrix(svm_pred, test_data$label)
svm_cm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  Broadleaf Canola Corn Oat Pea Soy Wheat
##   Broadleaf       167      1    0  11   0   0     2
##   Canola            5    192    0   2   2   0     0
##   Corn              8      0  197   1   0   0     4
##   Oat              15      2    0 155   1   1    37
##   Pea               1      4    0   1 195   0     1
##   Soy               1      1    2   0   2 195     8
##   Wheat             3      0    1  30   0   4   148
##
## Overall Statistics
##
##                Accuracy : 0.8921
##                  95% CI : (0.8747, 0.9079)
##     No Information Rate : 0.1429
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8742
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Broadleaf Class: Canola Class: Corn Class: Oat
## Sensitivity                    0.8350        0.9600      0.9850     0.7750
## Specificity                    0.9883        0.9925      0.9892     0.9533
## Pos Pred Value                 0.9227        0.9552      0.9381     0.7346
## Neg Pred Value                 0.9729        0.9933      0.9975     0.9622
## Prevalence                     0.1429        0.1429      0.1429     0.1429
## Detection Rate                 0.1193        0.1371      0.1407     0.1107
## Detection Prevalence           0.1293        0.1436      0.1500     0.1507
## Balanced Accuracy              0.9117        0.9763      0.9871     0.8642
##                      Class: Pea Class: Soy Class: Wheat
## Sensitivity              0.9750     0.9750       0.7400
## Specificity              0.9942     0.9883       0.9683
## Pos Pred Value           0.9653     0.9330       0.7957
## Neg Pred Value           0.9958     0.9958       0.9572
## Prevalence               0.1429     0.1429       0.1429
## Detection Rate           0.1393     0.1393       0.1057
## Detection Prevalence     0.1443     0.1493       0.1329
## Balanced Accuracy        0.9846     0.9817       0.8542
```

Train ANN

```
ann_model <- nnet(label ~ ., data = train_data, size = 5, maxit = 200)
```

```
## # weights:  92
## initial  value 12160.435302
## iter  10 value 10895.569312
## iter  20 value 10895.150757
## final  value 10895.150481
## converged
```

Ensure both are factors

```
ann_pred <- factor(ann_pred, levels = unique(test_data$label))
test_data$label <- factor(test_data$label, levels = unique(test_data$label))
```

Compute confusion matrix

```
length(ann_pred)

## [1] 21

length(test_data$label)

## [1] 1400

ann_pred <- predict(ann_model, newdata = test_data, type = "class")
levels_to_use <- sort(unique(c(levels(test_data$label), levels(ann_pred))))
ann_pred <- factor(ann_pred, levels = levels_to_use)
test_data$label <- factor(test_data$label, levels = levels_to_use)
any(is.na(ann_pred))  # TRUE if predictions contain NAs

## [1] FALSE

any(is.na(test_data$label))  # TRUE if labels contain NAs

## [1] FALSE

valid_indices <- complete.cases(ann_pred, test_data$label)
ann_pred <- ann_pred[valid_indices]
test_data$label <- test_data$label[valid_indices]
library(caret)
ann_cm <- confusionMatrix(ann_pred, test_data$label)
```

Create a summary table

```
results <- data.frame(
 Model = c("Decision Tree", "Random Forest", "SVM"),
 Accuracy = c(tree_cm$overall["Accuracy"], rf_cm$overall["Accuracy"],
svm_cm$overall["Accuracy"]),
 Kappa = c(tree_cm$overall["Kappa"], rf_cm$overall["Kappa"],
svm_cm$overall["Kappa"])
)
print(results)
```

```
##           Model  Accuracy     Kappa
## 1 Decision Tree 0.8050000 0.7725000
## 2 Random Forest 0.9057143 0.8900000
## 3           SVM 0.8921429 0.8741667
```

Unsupervized ML (Clustering)

Data Preparation

Exclude label column

```
data_features <- data[, -1] # Assuming the first column is 'label'
```

Standardize the data

```
data_scaled <- scale(data_features)
```

K-means clustering

```
set.seed(123) # For reproducibility
kmeans_result <- kmeans(data_scaled, centers = 7, nstart = 25) # 7 clusters
```

View cluster assignments

```
kmeans_clusters <- kmeans_result$cluster
```

Hierarchical Clustering

```
# Compute distance matrix
dist_matrix <- dist(data_scaled)
# Perform hierarchical clustering
hclust_result <- hclust(dist_matrix, method = "ward.D2")
# Cut the tree into 7 clusters
hclust_clusters <- cutree(hclust_result, k = 7)
```

Comparison

```
# Create a dataframe with actual labels and clustering results
comparison <- data.frame(
 Actual = data$label, # Original labels
 Kmeans = as.factor(kmeans_clusters),
 Hierarchical = as.factor(hclust_clusters)
)
# Install necessary library for ARI and NMI
library(flexclust)
```

```
## Loading required package: grid

## Loading required package: modeltools

## Loading required package: stats4

##
## Attaching package: 'flexclust'

## The following object is masked from 'package:e1071':
##
##     bclust
```

Calculate Adjusted Rand Index (ARI)

```
kmeans_ari <- randIndex(as.numeric(data$label), kmeans_clusters)
hclust_ari <- randIndex(as.numeric(data$label), hclust_clusters)
cat("K-means ARI:", kmeans_ari, "\n")

## K-means ARI: 0.4919438

cat("Hierarchical ARI:", hclust_ari, "\n")

## Hierarchical ARI: 0.4737311
```

Adjusted Rand Index (ARI)

Perform PCA

```
pca_result <- prcomp(data_scaled)
```

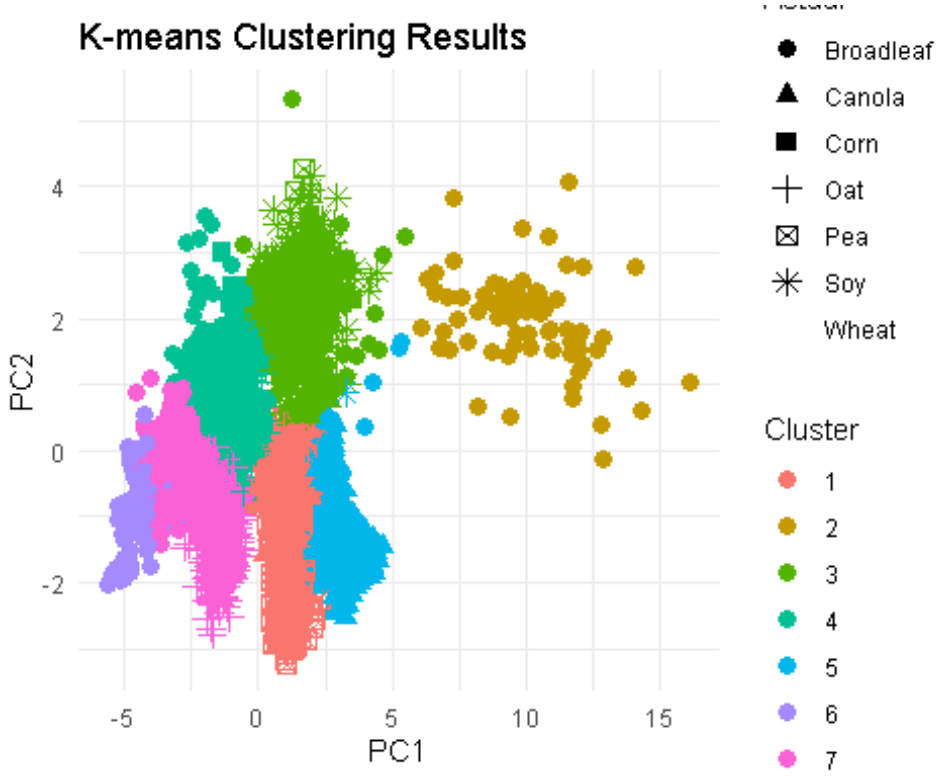
Create a dataframe for visualization

```
pca_data <- data.frame(
 PC1 = pca_result$x[, 1],
 PC2 = pca_result$x[, 2],
 Cluster = as.factor(kmeans_clusters),
 Actual = as.factor(data$label)
)
```

Plot PCA with clusters

```
ggplot(pca_data, aes(x = PC1, y = PC2, color = Cluster, shape = Actual)) +
 geom_point(size = 3) +
 labs(title = "K-means Clustering Results") +
 theme_minimal()

## Warning: The shape palette can deal with a maximum of 6 discrete values
because more
```

```
## than 6 becomes difficult to discriminate
## ℹ you have requested 7 values. Consider specifying shapes manually if
you need
##   that many have them.

## Warning: Removed 1000 rows containing missing values or values outside the
scale range
## (`geom_point()`).
```




```
table(Actual = data$label, Predicted = kmeans_clusters)

##            Predicted
## Actual        1   2   3   4   5   6   7
##   Broadleaf  15  76  38 182  13  92 584
##   Canola    109   0  14   1 875   0   1
##   Corn        0   0  80 856   0   0  64
##   Oat        10   0   8 253   0   0 729
##   Pea       966   0  32   2   0   0   0
##   Soy         0   0 987  12   1   0   0
##   Wheat       2   0  92 184   1   2 719
```