SMAProject

Creig

2024-11-15

```
library(readr)
library(readxl)

library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v purrr     1.0.2
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(ggplot2)

library(class)


#For Random Forest Algorithm
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
```

```
#For Support Vector Machine
library(e1071)
```

```
smadf<-read_excel("D:\\Minor Project\\Data\\SMARecoded.xlsx")

head(smadf)
```

```
## # A tibble: 6 x 22
##   Patientsage Gender City                State Diagnosisage Typeofdiagnostictest
##         <dbl> <chr>  <chr>               <chr>        <dbl> <chr>
## 1          10 Male   San Onofre          Sucre           10 MLPA
## 2          15 Female Chimichagua         Cesar           15 MLPA
## 3           9 Male   Sabaneta            Anti~            9 MLPA
## 4           7 Female Tunja               Boya~            7 MLPA
## 5           3 Female Cartagena de Indias Bolí~            3 MLPA
## 6           6 Female Pereira             Risa~            6 MLPA
## # i 16 more variables: Patientdiagnosisdate <dbl>, GeneticStudyResult <chr>,
## #   Mutationtype <chr>, SMAtype <chr>, SMN2copies <chr>, Gastrostomy <chr>,
## #   Tracheostomy <chr>, Locomotion <chr>, RGender <dbl>, RDiagnosticTest <dbl>,
## #   RGeneticStudy <dbl>, RMutationType <dbl>, RSMN2Copies <dbl>,
## #   RGastrostomy <dbl>, RTracheostomy <dbl>, RLocomotion <dbl>
```

Including Plots

You can also embed plots, for example:

```
attach(smadf)

smadata<-data.frame(Diagnosisage, RGender, RDiagnosticTest, RGeneticStudy, RMutationType, RSMN2Copies, R

head(smadata)
```

```
##   Diagnosisage RGender RDiagnosticTest RGeneticStudy RMutationType RSMN2Copies
## 1           10       1               1             2             1           4
## 2           15       2               1             1             2           3
## 3            9       1               1             1             3           3
## 4            7       2               1             1             3           3
## 5            3       2               1             1             2           4
## 6            6       2               1             1             2           4
##   RGastrostomy RTracheostomy RLocomotion SMAtype
## 1            0             0           4  Type 3
## 2            0             0           4  Type 3
## 3            0             0           3  Type 2
## 4            0             0           4  Type 3
## 5            0             0           4  Type 3
## 6            0             0           3  Type 2
```

```
smadata$SMAtype<-as.factor(smadata$SMAtype)
```

```
set.seed(385)

smashuf <- smadata[sample(nrow(smadata)),]

# Normalize the features (optional but recommended for KNN)
```

```r
normalize <- function(x) {
    return ((x - min(x)) / (max(x) - min(x)))
}

sma_norm <- as.data.frame(lapply(smadata[1:9], normalize))
sma_norm$SMAtype <- smadata$SMAtype
```

```r
# Split the data into training and test sets (70% train, 30% test)
train_index <- 1:161  # First 161 rows as training data

sma_train <- sma_norm[train_index, ]
sma_test <- sma_norm[-train_index, ]


train_labels <- smadata[train_index, 10]
test_labels <- smadata[-train_index, 10]
```

```r
# Set the value of k
k <- 5

# Train the KNN model and make predictions
knn_predictions <- knn(train = sma_train[, -10], test = sma_test[, -10], cl = train_labels, k = k)

# View the predictions
print(knn_predictions)
```

```
##  [1] Type 3 Type 4 Type 3 Type 1 Type 2 Type 2 Type 3 Type 2 Type 4 Type 1
## [11] Type 1 Type 1 Type 3 Type 3 Type 1 Type 1 Type 2 Type 2 Type 3 Type 3
## [21] Type 1 Type 4 Type 3 Type 1 Type 4 Type 3 Type 1 Type 1 Type 1 Type 4
## [31] Type 3 Type 2 Type 1 Type 3 Type 2 Type 3 Type 2 Type 2 Type 3 Type 3
## Levels: Type 1 Type 2 Type 3 Type 4
```

```r
# Create a confusion matrix
confusion_matrix <- table(knn_predictions, test_labels)

# Calculate accuracy
accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
print(paste("Accuracy:", accuracy))
```

```
## [1] "Accuracy: 0.65"
```

Random Forest Regression

```r
set.seed(405)

rf_model <- randomForest(SMAtype ~ ., data = smadata, ntree = 100)


rf_model
```

```
##
```

```
## Call:
##  randomForest(formula = SMAtype ~ ., data = smadata, ntree = 100)
##                Type of random forest: classification
##                      Number of trees: 100
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 6.47%
## Confusion matrix:
##        Type 1 Type 2 Type 3 Type 4 class.error
## Type 1     46      4      0      0  0.08000000
## Type 2      1     68      5      0  0.08108108
## Type 3      0      1     57      1  0.03389831
## Type 4      0      0      1     17  0.05555556
```

```
# Split the data into training and test sets
set.seed(123)
train_index <- 1:161 # 80% for training
train_data <- smadata[train_index, ]
test_data <- smadata[-train_index, ]
```

```
# Predict the test data
rf_predictions <- predict(rf_model, test_data)
```

```
# Create a confusion matrix
confusion_matrix <- table(Predicted = rf_predictions, Actual = test_data$SMAtype)
print(confusion_matrix)
```

```
##          Actual
## Predicted Type 1 Type 2 Type 3 Type 4
##    Type 1     14      0      0      0
##    Type 2      1      3      0      0
##    Type 3      0      1      9      0
##    Type 4      0      0      0     12
```

```
# Calculate accuracy
accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
print(paste("Accuracy: ", round(accuracy, 4)))
```

```
## [1] "Accuracy:  0.95"
```

```
# Get the OOB error estimate
oob_error <- rf_model$err.rate[nrow(rf_model$err.rate), "OOB"]
print(paste("OOB Error Rate: ", round(oob_error, 4)))
```

```
## [1] "OOB Error Rate:  0.0647"
```

Support Vector Machine

```
# Fit the SVM model
svm_model <- svm(SMAtype ~ Diagnosisage+RGender+RDiagnosticTest+RGeneticStudy+RMutationType+RSMN2Copies
                 data = smadata,
```

```
                  kernel = "linear",  # Using a linear kernel
                  cost = 1,           # Regularization parameter
                  scale = TRUE)       # Scale the features
```

```
# Predict the species
pred <- predict(svm_model, smadata)

# Display the first few predictions
head(pred)
```

```
##      1      2      3      4      5      6
## Type 3 Type 3 Type 2 Type 3 Type 3 Type 2
## Levels: Type 1 Type 2 Type 3 Type 4
```

```
# Confusion matrix to evaluate model accuracy
conf_matrix <- table(pred, smadata$SMAtype)
print(conf_matrix)
```

```
##
## pred     Type 1 Type 2 Type 3 Type 4
##   Type 1     47      2      0      0
##   Type 2      3     67      0      0
##   Type 3      0      5     59      0
##   Type 4      0      0      0     18
```

```
# Calculate accuracy
accuracy <- sum(pred == smadata$SMAtype) / length(smadata$SMAtype)
print(paste("Accuracy: ", round(accuracy * 100, 2), "%"))
```

```
## [1] "Accuracy:  95.02 %"
```

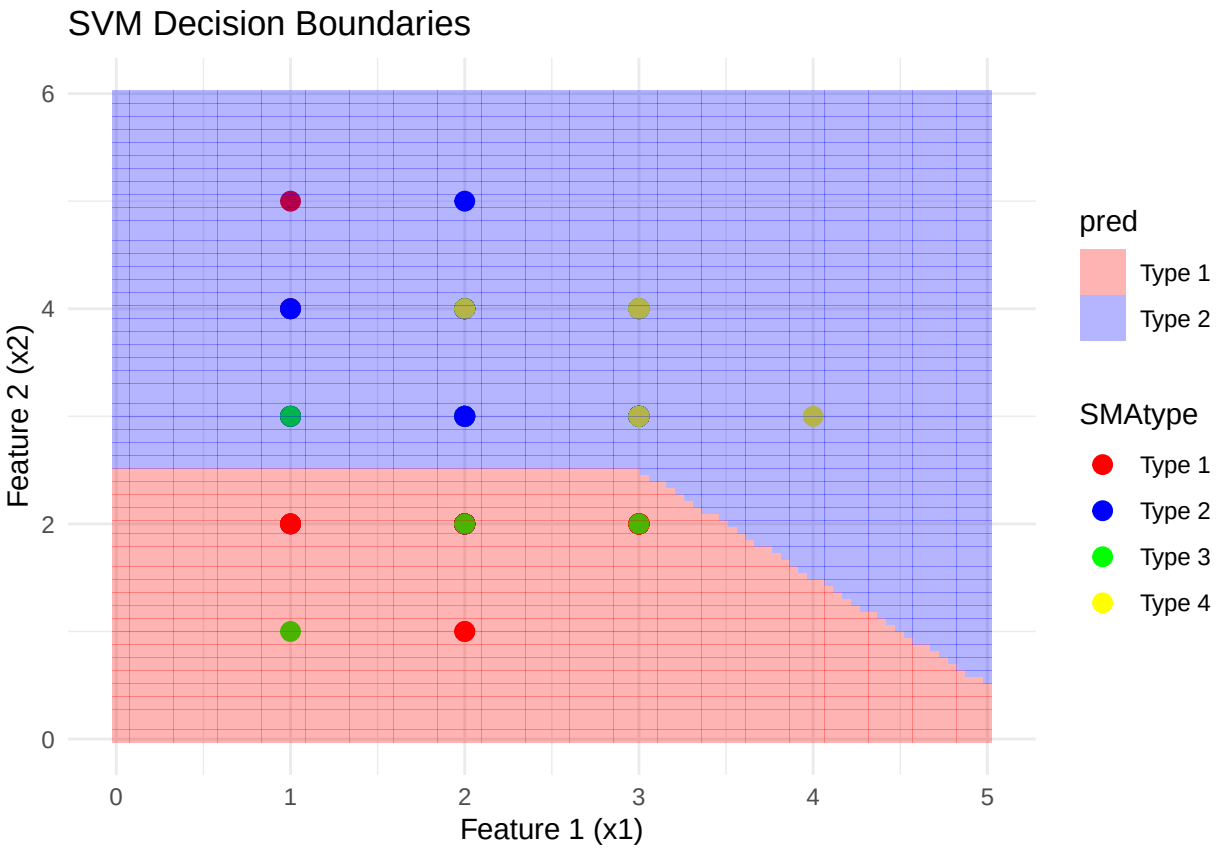
```
# Reduce the iris dataset to two features for visualization
sma_2d <- smadata[, c("RMutationType","RSMN2Copies","SMAtype")]

# Fit the SVM model on two features
svm_model_2d <- svm(SMAtype ~ RMutationType + RSMN2Copies, data=smadata, kernel = "linear")

# Create a grid of values to plot the decision boundaries
x_range <- seq(min(sma_2d$RMutationType) - 1, max(sma_2d$RMutationType) + 1, length.out = 100)
y_range <- seq(min(sma_2d$RSMN2Copies) - 1, max(sma_2d$RSMN2Copies) + 1, length.out = 100)
grid <- expand.grid(RMutationType = x_range, RSMN2Copies = y_range)

# Predict class for each point in the grid
grid$pred <- predict(svm_model_2d, grid)

# Step 4: Visualize the decision boundaries
ggplot(data=sma_2d, aes(x = RMutationType, y = RSMN2Copies)) +
  geom_point(aes(colour = SMAtype), size = 3) +
  geom_tile(data = grid, aes(x = RMutationType, y = RSMN2Copies, fill = pred), alpha = 0.3) +
  scale_color_manual(values = c("Type 1" = "red", "Type 2" = "blue", "Type 3" = "green", "Type 4" = "yel
  scale_fill_manual(values = c("Type 1" = "red", "Type 2" = "blue", "Type 3" = "green", "Type 4" = "yell
  theme_minimal() +
  labs(title = "SVM Decision Boundaries", x = "Feature 1 (x1)", y = "Feature 2 (x2)")
```


SVM Decision Boundaries
Feature 2 (x2)
Feature 1 (x1)
0
2
4
6
0
1
2
3
4
5
pred
Type 1
Type 2
SMAtype
Type 1
Type 2
Type 3
Type 4