

EXP 12: Ensemble Methods – Random Forest, GBM, and SVM on Sonar Dataset

AIM

Compare the performance of three widely used machine learning models - Random Forest (RF), Gradient Boosting Machine (GBM), and Support Vector Machine (SVM) - on the Sonar dataset, using R, as part of the Lab Migration Project.

EXPERIMENTAL SETUP

- Dataset: `Sonar` from the `mlbench` package
- Models: Random Forest (RF), Gradient Boosting Machine (GBM), Support Vector Machine (SVM)
- Data Split: 80% training, 20% testing
- Evaluation Metric: Accuracy using confusion matrix
- Validation Strategy: 10-fold repeated cross-validation (3 repeats)
- Visualization: Bar chart comparing model accuracies

LIBRARIES REQUIRED

```
library(caret)
```

```
## Loading required package: ggplot2
```

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

```
library(mlbench)
```

```
library(ggplot2)
```

```
# Load Sonar dataset
```

```
data(Sonar)
```

```
# View dataset structure
```

```
str(Sonar)
```

```

## 'data.frame':    208 obs. of  61 variables:
## $ V1 : num 0.02 0.0453 0.0262 0.01 0.0762 0.0286 0.0317 0.0519 0.0223 0.0164 ...
## $ V2 : num 0.0371 0.0523 0.0582 0.0171 0.0666 0.0453 0.0956 0.0548 0.0375 0.0173 ...
## $ V3 : num 0.0428 0.0843 0.1099 0.0623 0.0481 ...
## $ V4 : num 0.0207 0.0689 0.1083 0.0205 0.0394 ...
## $ V5 : num 0.0954 0.1183 0.0974 0.0205 0.059 ...
## $ V6 : num 0.0986 0.2583 0.228 0.0368 0.0649 ...
## $ V7 : num 0.154 0.216 0.243 0.11 0.121 ...
## $ V8 : num 0.16 0.348 0.377 0.128 0.247 ...
## $ V9 : num 0.3109 0.3337 0.5598 0.0598 0.3564 ...
## $ V10 : num 0.211 0.287 0.619 0.126 0.446 ...
## $ V11 : num 0.1609 0.4918 0.6333 0.0881 0.4152 ...
## $ V12 : num 0.158 0.655 0.706 0.199 0.395 ...
## $ V13 : num 0.2238 0.6919 0.5544 0.0184 0.4256 ...
## $ V14 : num 0.0645 0.7797 0.532 0.2261 0.4135 ...
## $ V15 : num 0.066 0.746 0.648 0.173 0.453 ...
## $ V16 : num 0.227 0.944 0.693 0.213 0.533 ...
## $ V17 : num 0.31 1 0.6759 0.0693 0.7306 ...
## $ V18 : num 0.3 0.887 0.755 0.228 0.619 ...
## $ V19 : num 0.508 0.802 0.893 0.406 0.203 ...
## $ V20 : num 0.48 0.782 0.862 0.397 0.464 ...
## $ V21 : num 0.578 0.521 0.797 0.274 0.415 ...
## $ V22 : num 0.507 0.405 0.674 0.369 0.429 ...
## $ V23 : num 0.433 0.396 0.429 0.556 0.573 ...
## $ V24 : num 0.555 0.391 0.365 0.485 0.54 ...
## $ V25 : num 0.671 0.325 0.533 0.314 0.316 ...
## $ V26 : num 0.641 0.32 0.241 0.533 0.229 ...
## $ V27 : num 0.71 0.327 0.507 0.526 0.7 ...
## $ V28 : num 0.808 0.277 0.853 0.252 1 ...
## $ V29 : num 0.679 0.442 0.604 0.209 0.726 ...
## $ V30 : num 0.386 0.203 0.851 0.356 0.472 ...
## $ V31 : num 0.131 0.379 0.851 0.626 0.51 ...
## $ V32 : num 0.26 0.295 0.504 0.734 0.546 ...
## $ V33 : num 0.512 0.198 0.186 0.612 0.288 ...
## $ V34 : num 0.7547 0.2341 0.2709 0.3497 0.0981 ...
## $ V35 : num 0.854 0.131 0.423 0.395 0.195 ...
## $ V36 : num 0.851 0.418 0.304 0.301 0.418 ...
## $ V37 : num 0.669 0.384 0.612 0.541 0.46 ...
## $ V38 : num 0.61 0.106 0.676 0.881 0.322 ...
## $ V39 : num 0.494 0.184 0.537 0.986 0.283 ...
## $ V40 : num 0.274 0.197 0.472 0.917 0.243 ...
## $ V41 : num 0.051 0.167 0.465 0.612 0.198 ...
## $ V42 : num 0.2834 0.0583 0.2587 0.5006 0.2444 ...
## $ V43 : num 0.282 0.14 0.213 0.321 0.185 ...
## $ V44 : num 0.4256 0.1628 0.2222 0.3202 0.0841 ...
## $ V45 : num 0.2641 0.0621 0.2111 0.4295 0.0692 ...
## $ V46 : num 0.1386 0.0203 0.0176 0.3654 0.0528 ...
## $ V47 : num 0.1051 0.053 0.1348 0.2655 0.0357 ...
## $ V48 : num 0.1343 0.0742 0.0744 0.1576 0.0085 ...
## $ V49 : num 0.0383 0.0409 0.013 0.0681 0.023 0.0264 0.0507 0.0285 0.0777 0.0092 ...
## $ V50 : num 0.0324 0.0061 0.0106 0.0294 0.0046 0.0081 0.0159 0.0178 0.0439 0.0198 ...
## $ V51 : num 0.0232 0.0125 0.0033 0.0241 0.0156 0.0104 0.0195 0.0052 0.0061 0.0118 ...
## $ V52 : num 0.0027 0.0084 0.0232 0.0121 0.0031 0.0045 0.0201 0.0081 0.0145 0.009 ...
## $ V53 : num 0.0065 0.0089 0.0166 0.0036 0.0054 0.0014 0.0248 0.012 0.0128 0.0223 ...

```

```
## $ V54 : num 0.0159 0.0048 0.0095 0.015 0.0105 0.0038 0.0131 0.0045 0.0145 0.0179 ...
## $ V55 : num 0.0072 0.0094 0.018 0.0085 0.011 0.0013 0.007 0.0121 0.0058 0.0084 ...
## $ V56 : num 0.0167 0.0191 0.0244 0.0073 0.0015 0.0089 0.0138 0.0097 0.0049 0.0068 ...
## $ V57 : num 0.018 0.014 0.0316 0.005 0.0072 0.0057 0.0092 0.0085 0.0065 0.0032 ...
## $ V58 : num 0.0084 0.0049 0.0164 0.0044 0.0048 0.0027 0.0143 0.0047 0.0093 0.0035 ...
## $ V59 : num 0.009 0.0052 0.0095 0.004 0.0107 0.0051 0.0036 0.0048 0.0059 0.0056 ...
## $ V60 : num 0.0032 0.0044 0.0078 0.0117 0.0094 0.0062 0.0103 0.0053 0.0022 0.004 ...
## $ Class: Factor w/ 2 levels "M","R": 2 2 2 2 2 2 2 2 2 2 ...
```

```
set.seed(123)
trainIndex <- createDataPartition(Sonar$Class, p = 0.8, list = FALSE)
data_train <- Sonar[trainIndex, ]
data_test <- Sonar[-trainIndex, ]
```

```
ctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 3)
```

```
set.seed(123)
rf_model <- train(Class ~ ., data = data_train,
                  method = "rf",
                  trControl = ctrl)
```

```
set.seed(123)
gbm_model <- train(Class ~ ., data = data_train,
                  method = "gbm",
                  trControl = ctrl,
                  verbose = FALSE)
```

```
set.seed(123)
svm_model <- train(Class ~ ., data = data_train,
                  method = "svmRadial",
                  trControl = ctrl)
```

```
# Predictions
rf_predictions <- predict(rf_model, data_test)
gbm_predictions <- predict(gbm_model, data_test)
svm_predictions <- predict(svm_model, data_test)

# Accuracy calculation
rf_accuracy <- confusionMatrix(rf_predictions, data_test$Class)$overall["Accuracy"]
gbm_accuracy <- confusionMatrix(gbm_predictions, data_test$Class)$overall["Accuracy"]
svm_accuracy <- confusionMatrix(svm_predictions, data_test$Class)$overall["Accuracy"]

# Print accuracies
cat("Random Forest Accuracy:", rf_accuracy, "\n")
```

```
## Random Forest Accuracy: 0.9512195
```

```
cat("GBM Accuracy:", gbm_accuracy, "\n")
```

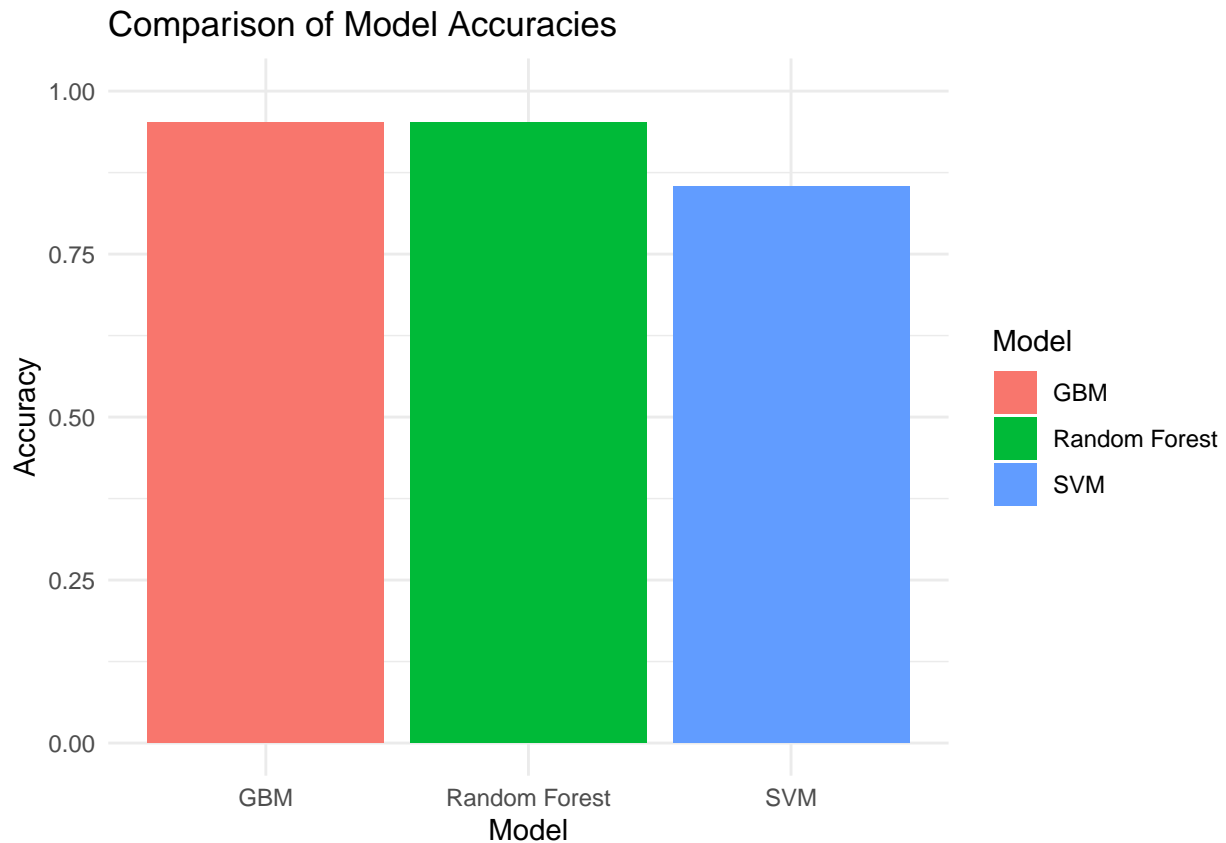
```
## GBM Accuracy: 0.9512195
```

```
cat("SVM Accuracy:", svm_accuracy, "\n")
```

```
## SVM Accuracy: 0.8536585
```

```
# Combine into data frame
model_accuracies <- data.frame(
  Model = c("Random Forest", "GBM", "SVM"),
  Accuracy = c(rf_accuracy, gbm_accuracy, svm_accuracy)
)

# Plot
ggplot(model_accuracies, aes(x = Model, y = Accuracy, fill = Model)) +
  geom_bar(stat = "identity") +
  ylim(0, 1) +
  labs(title = "Comparison of Model Accuracies",
       x = "Model",
       y = "Accuracy") +
  theme_minimal()
```



CONCLUSION

The conclusions drawn from this analysis can be used to determine the most suitable machine learning model for the Sonar dataset, which can be further applied to real-world classification problems with similar characteristics.