Data 622 Machine Learning and Big Data_HW3

Jiaxin Zheng

2025-11-05

Part I

1.1 Assignment Introduction:

- Read the following articles: https://www.hindawi.com/journals/complexity/2021/5550344/ https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8137961/
- Search for academic content (at least 3 articles) that compare the use of decision trees vs SVMs in your current area of expertise.
- Perform an analysis of the dataset used in Homework #2 using the SVM algorithm.
- Compare the results with the results from previous homework.
- Answer questions, such as:
  - Which algorithm is recommended to get more accurate results?
  - Is it better for classification or regression scenarios?
  - Do you agree with the recommendations?
  - Why?

Part II: Pre-processing

1.1 Load the csv file and nessary libraries.

```
# Load Libraries
library(caret)
library(rpart)
library(rpart.plot)
library(randomForest)
library(pROC)
library(ada)
library(tidyverse)
library(adabag)
library(corrplot)
library(dplyr)
library(knitr)
library(skimr)
library(readr)
library(kernlab)
library(e1071)
```

```
# Read data file
df <- read.csv("https://raw.githubusercontent.com/Jennyjjxxzz/HW1/refs/heads/main/bank-full.csv", sep =
```

```r
head (df)
```

```
##   age           job marital education default balance housing loan contact day
## 1  58    management married  tertiary      no    2143     yes   no unknown   5
## 2  44    technician  single secondary      no      29     yes   no unknown   5
## 3  33 entrepreneur married secondary      no       2     yes  yes unknown   5
## 4  47   blue-collar married   unknown      no    1506     yes   no unknown   5
## 5  33       unknown  single   unknown      no       1      no   no unknown   5
## 6  35    management married  tertiary      no     231     yes   no unknown   5
##   month duration campaign pdays previous poutcome  y
## 1   may      261        1    -1        0  unknown no
## 2   may      151        1    -1        0  unknown no
## 3   may       76        1    -1        0  unknown no
## 4   may       92        1    -1        0  unknown no
## 5   may      198        1    -1        0  unknown no
## 6   may      139        1    -1        0  unknown no
```

1.2 Convert the variables to factor

```r
cat_cols <- c("job","marital","education","default","housing","loan",
              "contact","month","poutcome","y")
num_cols <- c("age","balance","day","duration","campaign","pdays","previous")

df <- df %>%
  mutate(
    across(all_of(cat_cols), ~ as.factor(.x)),
    across(all_of(num_cols), ~ as.numeric(.x))
  )
```

Part III: Split the data 80 training/ 20 testing

```r
set.seed(123)
# Slit the data (80% training, 20% testing)
idx   <- createDataPartition(df$y, p = 0.8, list = FALSE)
train <- df[idx, ]
test  <- df[-idx, ]
```

Part IV: Experiment

3.1 SVM Linear:

```r
# Train a linear SVM using svmLinear
set.seed(123)

svm_model_linear <- svm(
  y ~ .,
  data = train,
  kernal = "linear",
  probability = TRUE,
  scale = TRUE
)
```

```r
# Predict
svm_pred_prob <- attr(predict(svm_model_linear, test, probability = TRUE), "probabilities")[, "yes"]
svm_pred_class <- ifelse(svm_pred_prob > 0.5, "yes", "no")
svm_pred_class <- factor(svm_pred_class, levels = c("no", "yes"))

# Confusion matrix
cm_svm_linear <- confusionMatrix(svm_pred_class, test$y, positive = "yes")
print(cm_svm_linear)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   no  yes
##        no  7827  766
##        yes  157  291
##
##                Accuracy : 0.8979
##                  95% CI : (0.8915, 0.9041)
##     No Information Rate : 0.8831
##     P-Value [Acc > NIR] : 4.297e-06
##
##                   Kappa : 0.3408
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.27531
##             Specificity : 0.98034
##          Pos Pred Value : 0.64955
##          Neg Pred Value : 0.91086
##              Prevalence : 0.11691
##          Detection Rate : 0.03219
##    Detection Prevalence : 0.04955
##       Balanced Accuracy : 0.62782
##
##        'Positive' Class : yes
##
```

```r
precision_svm <- cm_svm_linear$byClass["Pos Pred Value"]
recall_svm    <- cm_svm_linear$byClass["Sensitivity"]
f1_svm        <- (2 * precision_svm * recall_svm) / (precision_svm + recall_svm)

roc_svm <- roc(response = factor(test$y, levels = c("no", "yes")),
               predictor = svm_pred_prob)
```

```
## Setting levels: control = no, case = yes
```

```
## Setting direction: controls < cases
```

```r
auc_svm <- as.numeric(auc(roc_svm))

# Combine metrics into a clean table
results_svm_linear_base <- data.frame(
  Model = "SVM Linear - Baseline",
  Accuracy  = cm_svm_linear$overall["Accuracy"],
  Precision = precision_svm,
```

```
  Recall    = recall_svm,
  F1_Score  = f1_svm,
  AUC       = auc_svm
)

kable(results_svm_linear_base, caption = "SVM Linear Baseline Results")
```

Table 1: SVM Linear Baseline Results

	Model	Accuracy	Precision	Recall	F1_Score	AUC
Accuracy	SVM Linear - Baseline	0.8979095	0.6495536	0.2753075	0.386711	0.9040328

3.2 SVM Linear Tuned:

```
set.seed(123)
svm_linear_tuned <- tune.svm(
  y ~ ., data = train,
  kernel = "linear",
  cost = c(0.001, 0.01, 0.1),
  probability = TRUE,
  tunecontrol = tune.control(cross = 5)
)
```

```
# Use best fitted model directly
svm_linear_tuned <- svm_linear_tuned$best.model
```

```
# Predict
svm_linear_tuned_pred  <- predict(svm_linear_tuned, test, probability = TRUE)
svm_linear_tuned_probs <- attr(svm_linear_tuned_pred, "probabilities")[, "yes"]
svm_linear_tuned_pred  <- factor(ifelse(svm_linear_tuned_probs >= 0.5, "yes", "no"),
                                 levels = levels(test$y))
```

```
# Metrics
cm   <- confusionMatrix(svm_linear_tuned_pred, test$y, positive = "yes")
prec <- cm$byClass["Pos Pred Value"]; rec <- cm$byClass["Sensitivity"]
f1   <- (2 * prec * rec) / (prec + rec)
aucv <- as.numeric(auc(roc(test$y, svm_linear_tuned_probs, levels = rev(levels(test$y)))))
```

```
## Setting direction: controls > cases
```

```
# Combine metrics into a clean table
results_svm_linear_tuned <- data.frame(
  Model     = "SVM Linear - Tuned",
  Accuracy  = cm$overall["Accuracy"],
  Precision = prec, Recall = rec,
  F1_Score  = f1, AUC = aucv
)

kable(results_svm_linear_tuned, caption = "SVM Linear (Tuned) Results")
```

Table 2: SVM Linear (Tuned) Results

	Model	Accuracy	Precision	Recall	F1_Score	AUC
Accuracy	SVM Linear - Tuned	0.8926004	0.6524823	0.1740776	0.274832	0.8970666

3.3 SVM Radial:

```
svm_rad <- svm(formula = y ~ .,
               data = train,
               kernel = "radial",
               probability = TRUE,
               scale = TRUE)
```

```
svm_rad_prob <- predict(svm_rad, test, probability = TRUE)
svm_rad_prob_num <- as.numeric(attr(svm_rad_prob, "probabilities")[, "yes"])
svm_rad_pred <- factor(ifelse(svm_rad_prob_num >= 0.5, "yes", "no"),
                       levels = levels(test$y))
```

```
# confusion matrix
cm_svm_rad <- confusionMatrix(svm_rad_pred, test$y, positive = "yes")
print(cm_svm_rad)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   no  yes
##        no  7826  766
##        yes  158  291
##
##                Accuracy : 0.8978
##                  95% CI : (0.8914, 0.904)
##     No Information Rate : 0.8831
##     P-Value [Acc > NIR] : 5.031e-06
##
##                   Kappa : 0.3405
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.27531
##             Specificity : 0.98021
##          Pos Pred Value : 0.64811
##          Neg Pred Value : 0.91085
##              Prevalence : 0.11691
##          Detection Rate : 0.03219
##    Detection Prevalence : 0.04966
##       Balanced Accuracy : 0.62776
##
##        'Positive' Class : yes
##
```

```
# metrics
precision_rad <- cm_svm_rad$byClass["Pos Pred Value"]
recall_rad    <- cm_svm_rad$byClass["Sensitivity"]
f1_rad        <- (2 * precision_rad * recall_rad) / (precision_rad + recall_rad)
```

```r
roc_rad <- roc(response = test$y,
               predictor = svm_rad_prob_num,
               levels = rev(levels(test$y)))
```

```
## Setting direction: controls > cases
```

```r
auc_rad <- as.numeric(auc(roc_rad))

# result
results_svm_rad <- data.frame(
  Model     = "SVM - Radial (baseline)",
  Accuracy  = cm_svm_rad$overall["Accuracy"],
  Precision = precision_rad,
  Recall    = recall_rad,
  F1_Score  = f1_rad,
  AUC       = auc_rad
)
kable(results_svm_rad, caption = "SVM Radial (Baseline) Results")
```

Table 3: SVM Radial (Baseline) Results

	Model	Accuracy	Precision	Recall	F1_Score	AUC
Accuracy	SVM - Radial (baseline)	0.8977989	0.6481069	0.2753075	0.3864542	0.9040328

3.4 SVM Radial Tuned:

```r
ctrl_cv <- trainControl(
  method = "cv", number = 5,
  classProbs = TRUE,
  summaryFunction = twoClassSummary,
  savePredictions = "final"
)
```

```r
set.seed(123)

svm_rbf_tuned <- e1071::tune.svm(
  y ~ ., data = train,
  kernel = "radial",
  probability = TRUE,
  scale = TRUE,,
  trControl = ctrl_cv,
  preProcess = c("center","scale"),
  tuneGrid = expand.grid(.sigma= c(0.001, 0.01, 0.1, 1, 5, 10),
                         .C=c(0.001, 0.01, 0.1, 1, 5, 10))
)
```

```r
# Use the best fitted model
best_rbf <- svm_rbf_tuned$best.model
```

```r
# Predict
rbf_pred_obj <- predict(best_rbf, newdata = test, probability = TRUE)
svm_rbf_prob <- attr(rbf_pred_obj, "probabilities")[, "yes"]
svm_rbf_pred <- factor(ifelse(svm_rbf_prob >= 0.5, "yes", "no"),
                       levels = levels(test$y))
```

```r
# Metrics
cm_rbf <- caret::confusionMatrix(svm_rbf_pred, test$y, positive = "yes")
precision_rbf <- cm_rbf$byClass["Pos Pred Value"]
recall_rbf    <- cm_rbf$byClass["Sensitivity"]
f1_rbf        <- (2 * precision_rbf * recall_rbf) / (precision_rbf + recall_rbf)

roc_rbf <- pROC::roc(response = test$y, predictor = svm_rbf_prob, levels = rev(levels(test$y)))
```

```
## Setting direction: controls > cases
```

```r
auc_rbf <- as.numeric(pROC::auc(roc_rbf))

results_svm_rbf <- data.frame(
  Model     = "SVM - Radial (tuned)",
  Accuracy  = cm_rbf$overall["Accuracy"],
  Precision = precision_rbf,
  Recall    = recall_rbf,
  F1_Score  = f1_rbf,
  AUC       = auc_rbf
)

kable(results_svm_rbf, caption = "SVM Radial (Tuned) Results")
```

Table 4: SVM Radial (Tuned) Results

	Model	Accuracy	Precision	Recall	F1_Score	AUC
Accuracy	SVM - Radial (tuned)	0.8977989	0.6481069	0.2753075	0.3864542	0.9040328

3.5 SVM Polynomial:

```r
svm_polynomial_prob <- svm(
  y ~ ., data = train,
  kernel = "polynomial",
  probability = TRUE, scale = TRUE
)
```

```r
# Predict
svm_polynomial_pred0 <- predict(svm_polynomial_prob, newdata = test, probability = TRUE)
svm_polynomial_probs <- attr(svm_polynomial_pred0, "probabilities")[, "yes"]
svm_polynomial_pred  <- factor(ifelse(svm_polynomial_probs >= 0.5, "yes", "no"),
                               levels = levels(test$y))
```

```r
# Metrics
confusion_matrix_svm_polynomial <- confusionMatrix(svm_polynomial_pred, test$y, positive = "yes")
precision_poly <- confusion_matrix_svm_polynomial$byClass["Pos Pred Value"]
recall_poly    <- confusion_matrix_svm_polynomial$byClass["Sensitivity"]
f1_poly        <- (2 * precision_poly * recall_poly) / (precision_poly + recall_poly)

roc_svm_polynomial <- roc(response = test$y,
                          predictor = svm_polynomial_probs,
                          levels = rev(levels(test$y)))
```

```
## Setting direction: controls > cases
```

```r
auc_svm_polynomial <- as.numeric(auc(roc_svm_polynomial))

# Result row
results_svm_polynomial <- data.frame(
  Model     = "SVM - Polynomial (baseline)",
  Accuracy  = confusion_matrix_svm_polynomial$overall["Accuracy"],
  Precision = precision_poly,
  Recall    = recall_poly,
  F1_Score  = f1_poly,
  AUC       = auc_svm_polynomial
)

kable(results_svm_polynomial, caption = "SVM Polynomial (Baseline) Results")
```

Table 5: SVM Polynomial (Baseline) Results

	Model	Accuracy	Precision	Recall	F1_Score	AUC
Accuracy	SVM - Polynomial (baseline)	0.8933746	0.6666667	0.1759697	0.2784431	0.8829763

```r
standardize_results <- function(df) {
  wanted <- c("Model", "Accuracy", "Precision", "Recall", "F1_Score", "AUC")
  if (!"AUC" %in% names(df)) df$AUC <- NA_real_
  df %>%
    dplyr::mutate(
      Model     = as.character(Model),
      Accuracy  = as.numeric(Accuracy),
      Precision = as.numeric(Precision),
      Recall    = as.numeric(Recall),
      F1_Score  = as.numeric(F1_Score),
      AUC       = as.numeric(AUC)
    ) %>%
    dplyr::select(dplyr::all_of(wanted))
}

# Order
svm_results_list <- list(
  results_svm_linear_base,   # 1. Linear Baseline
  results_svm_linear_tuned,  # 2. Linear Tuned
  results_svm_rad,           # 3. Radial Baseline
  results_svm_rbf,           # 4. Radial Tuned
  results_svm_polynomial     # 5. Polynomial Baseline
)

svm_results_table <- svm_results_list %>%
  lapply(standardize_results) %>%
  dplyr::bind_rows() %>%
  dplyr::mutate(dplyr::across(where(is.numeric), ~ round(., 4)))

# Display in table
knitr::kable(
  svm_results_list,
  caption = "SVM Model Comparison: Linear, Linear Tuned, Radial, Radial Tuned, Polynomial"
```

Table 6: SVM Model Comparison: Linear, Linear Tuned, Radial, Radial Tuned, Polynomial

	Model	Accuracy	Precision	Recall	F1_Score	AUC
Accuracy	SVM Linear - Baseline	0.8979095	0.6495536	0.2753075	0.386711	0.9040328

	Model	Accuracy	Precision	Recall	F1_Score	AUC
Accuracy	SVM Linear - Tuned	0.8926004	0.6524823	0.1740776	0.274832	0.8970666

	Model	Accuracy	Precision	Recall	F1_Score	AUC
Accuracy	SVM - Radial (baseline)	0.8977989	0.6481069	0.2753075	0.3864542	0.9040328

	Model	Accuracy	Precision	Recall	F1_Score	AUC
Accuracy	SVM - Radial (tuned)	0.8977989	0.6481069	0.2753075	0.3864542	0.9040328

	Model	Accuracy	Precision	Recall	F1_Score	AUC
Accuracy	SVM - Polynomial (baseline)	0.8933746	0.6666667	0.1759697	0.2784431	0.8829763

```
)
```