📊 Creditworthiness Prediction Analysis

Creditworthiness Prediction using CART and Random Forest

Code

Khairunnisa Maharani

2025-06-07

📊 Creditworthiness Prediction Analysis

This project explores the prediction of creditworthiness using machine learning, specifically Decision Tree (CART) and Random Forest algorithms. It handles class imbalance using ROSE and compares model performance with various configurations.

🗂️ Dataset Description

- **Dataset**: German Credit Dataset from UCI ML Repository
- **Samples**: 1000 customers
- **Target Variable**: Creditworthiness (`Good` or `Bad`)
- **Features**: 20 predictor variables (demographic and financial attributes)

---

1. Load Required Libraries

Hide

```
# Core data manipulation and visualization
library(tidyverse)
library(ggplot2)
library(gridExtra)
library(corrplot)
library(knitr)

# Machine learning libraries
library(rpart)
library(rpart.plot)
library(randomForest)
library(caret)

# Model evaluation and handling imbalanced data
library(ROSE)
library(pROC)

# Additional visualization and analysis
library(visdat)
library(naniar)
library(skimr)
library(DT)

# Set theme for consistent plots
theme_set(theme_minimal())
```

2. Data Loading and Initial Exploration

Hide

```
# Load dataset
data <- read.csv("german_data.csv")

# Basic data exploration
cat("Dataset dimensions:", dim(data), "\n")
```

```
## Dataset dimensions: 1000 21
```

Hide

```
cat("Dataset structure:\n")
```

```
## Dataset structure:
```

Hide

```
str(data)
```

```
## 'data.frame':    1000 obs. of  21 variables:
##  $ Status.Rekening                : chr  "< 0 DM" "0-199 DM" "Tidak ada rekenin
g" "< 0 DM" ...
##  $ Durasi.dalam.bulan             : int  6 48 12 42 24 36 24 36 12 30 ...
##  $ Riwayat.Kredit                 : chr  "Masalah / kredit di luar bank" "Kredit
berjalan lancar" "Masalah / kredit di luar bank" "Kredit berjalan lancar" ...
##  $ Tujuan.Kredit                  : chr  "TV / Radio" "TV / Radio" "Pendidikan"
"Furniture" ...
##  $ Jumlah.Kredit..DM.             : int  1169 5951 2096 7882 4870 9055 2835 6948
3059 5234 ...
##  $ Tabungan.Bonds                 : chr  "Tidak ada / tidak tahu" "< 100 DM" "<
100 DM" "< 100 DM" ...
##  $ Pekerjaan.Saat.Ini             : chr  ">= 7 thn" "1-4 thn" "4-7 thn" "4-7 th
n" ...
##  $ Tingkat.Angsuran....           : int  4 2 2 2 3 2 3 2 2 4 ...
##  $ Status.Pribadi.dan.Jenis.Kelamin: chr  "Pria lajang" "Wanita menikah/cerai" "P
ria lajang" "Pria lajang" ...
##  $ Debitur...Penjamin             : chr  "Tidak ada" "Tidak ada" "Tidak ada" "Pe
njamin" ...
##  $ Tinggal.di.Tempat.Saat.Ini     : int  4 2 3 4 4 4 4 2 4 2 ...
##  $ Properti                       : chr  "Properti" "Properti" "Properti" "Tabun
gan/Asuransi" ...
##  $ Usia..tahun.                   : int  67 22 49 45 53 35 53 35 61 28 ...
##  $ Rencana.Angsuran.Lain          : chr  "Tidak ada" "Tidak ada" "Tidak ada" "Ti
dak ada" ...
##  $ Tempat.Tinggal                 : chr  "Milik sendiri" "Milik sendiri" "Milik
sendiri" "Gratis" ...
##  $ Jumlah.Kredit.di.Bank.Ini      : int  2 1 1 1 2 1 1 1 1 2 ...
##  $ Pekerjaan                      : chr  "Terampil / pegawai" "Terampil / pegawa
i" "Tidak terampil (residen)" "Terampil / pegawai" ...
##  $ Jumlah.Tanggungan              : int  1 1 2 2 2 2 1 1 1 1 ...
##  $ Nomor.Telepon                  : chr  "Ada" "Tidak ada" "Tidak ada" "Tidak ad
a" ...
##  $ Pekerja.Migran                 : chr  "Ya" "Ya" "Ya" "Ya" ...
##  $ Kelayakan.Kredit               : chr  "Baik" "Buruk" "Baik" "Baik" ...
```

Hide

```
# Comprehensive data summary using skimr
skim_result <- skim(data)
print(skim_result)
```

```
## ── Data Summary ────────────────────────
##                            Values
## Name                       data
## Number of rows             1000
## Number of columns          21
## _________________________
## Column type frequency:
##   character                14
##   numeric                  7
## _________________________
## Group variables            None
##
## ── Variable type: character ─────────────────────────────────────────────────
##    skim_variable                    n_missing complete_rate min max empty
##  1 Status.Rekening                          0             1   6  22     0
##  2 Riwayat.Kredit                           0             1  18  35     0
##  3 Tujuan.Kredit                            0             1   6  15     0
##  4 Tabungan.Bonds                           0             1   8  22     0
##  5 Pekerjaan.Saat.Ini                       0             1   7  10     0
##  6 Status.Pribadi.dan.Jenis.Kelamin         0             1  11  20     0
##  7 Debitur...Penjamin                       0             1   8  10     0
##  8 Properti                                 0             1   8  22     0
##  9 Rencana.Angsuran.Lain                    0             1   4   9     0
## 10 Tempat.Tinggal                           0             1   4  13     0
## 11 Pekerjaan                                0             1  18  43     0
## 12 Nomor.Telepon                            0             1   3   9     0
## 13 Pekerja.Migran                           0             1   2   5     0
## 14 Kelayakan.Kredit                         0             1   4   5     0
##    n_unique whitespace
##  1        4          0
##  2        5          0
##  3       10          0
##  4        5          0
##  5        5          0
##  6        4          0
##  7        3          0
##  8        4          0
##  9        3          0
## 10        3          0
## 11        4          0
## 12        2          0
## 13        2          0
## 14        2          0
##
## ── Variable type: numeric ───────────────────────────────────────────────────
##   skim_variable              n_missing complete_rate    mean       sd  p0   p25
## 1 Durasi.dalam.bulan                 0             1   20.9    12.1     4   12
## 2 Jumlah.Kredit..DM.                 0             1 3271.   2823.    250 1366.
```

```
## 3 Tingkat.Angsuran....                0             1    2.97    1.12    1    2
## 4 Tinggal.di.Tempat.Saat.Ini          0             1    2.84    1.10    1    2
## 5 Usia..tahun.                        0             1   35.5    11.4    19   27
## 6 Jumlah.Kredit.di.Bank.Ini           0             1    1.41    0.578   1    1
## 7 Jumlah.Tanggungan                   0             1    1.16    0.362   1    1
##     p50   p75  p100 hist
## 1   18    24     72 ▇▇▃▁▁
## 2 2320. 3972. 18424 ▇▂▁▁▁
## 3    3     4      4 ▂▂▁▂▇
## 4    3     4      4 ▂▇▁▂▇
## 5   33    42     75 ▇▇▃▁▁
## 6    1     2      4 ▇▇▁▁▁
## 7    1     1      2 ▇▁▁▁▂
```

3. Data Quality Assessment

Hide

```
# Check for missing values
missing_values <- colSums(is.na(data))
missing_percentage <- (missing_values / nrow(data)) * 100
missing_df <- data.frame(
    Missing_Count = missing_values,
    Missing_Percentage = missing_percentage
)

# Display missing values if any
if(sum(missing_values) > 0) {
    cat("Missing Values Summary:\n")
    print(missing_df[missing_df$Missing_Count > 0, ])

    # Visualize missing values pattern
    vis_miss(data, cluster = TRUE)
    gg_miss_var(data)
} else {
    cat("✅ No missing values found in the dataset!\n")
}
```

```
## ✅ No missing values found in the dataset!
```

Hide

```
# Convert categorical variables to factors
categorical_vars <- names(data)[sapply(data, is.character)]
for(var in categorical_vars) {
    data[[var]] <- as.factor(data[[var]])
}

# Convert target variable to factor with proper levels
data$Kelayakan.Kredit <- factor(data$Kelayakan.Kredit, levels = c("Buruk", "Baik"))

cat("Categorical variables converted:", length(categorical_vars), "variables\n")
```

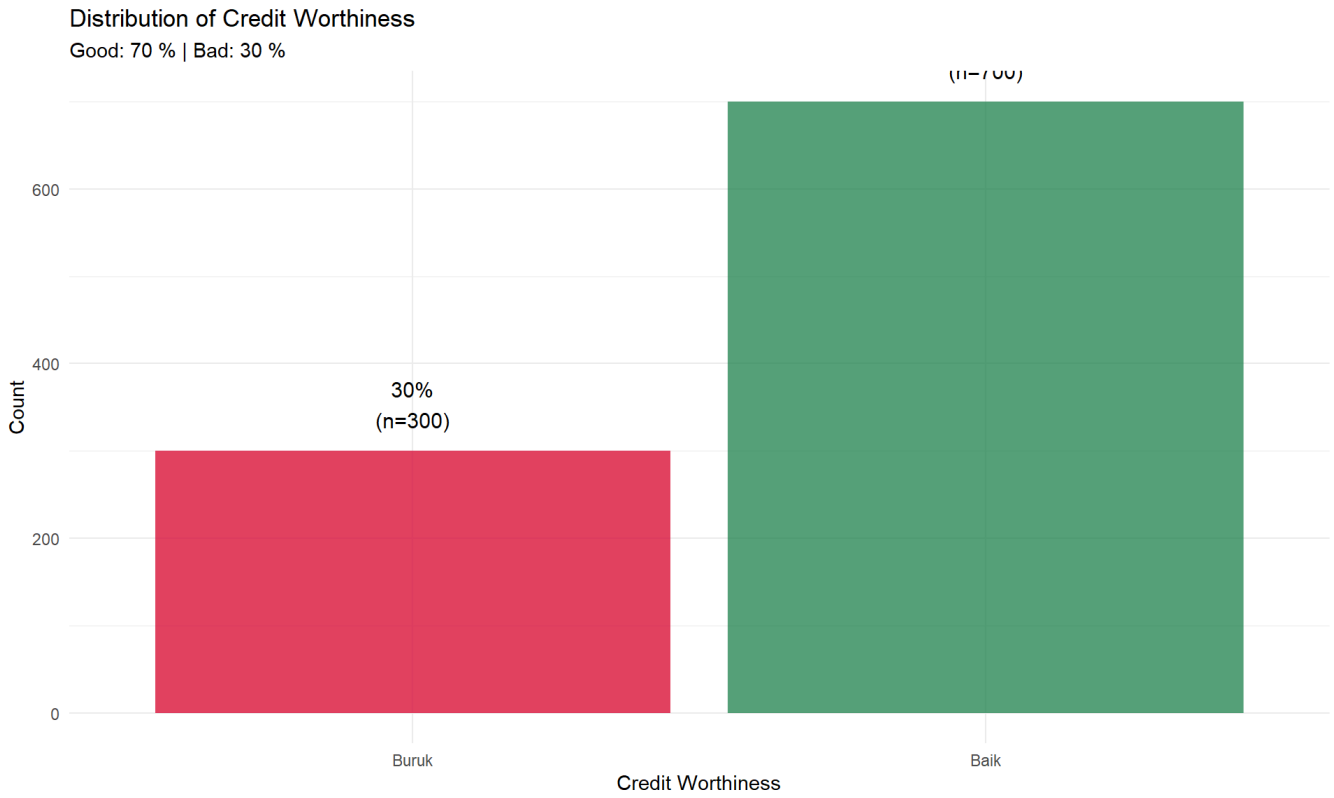
```
## Categorical variables converted: 14 variables
```

4. Exploratory Data Analysis (EDA)

4.1 Target Variable Distribution

Hide

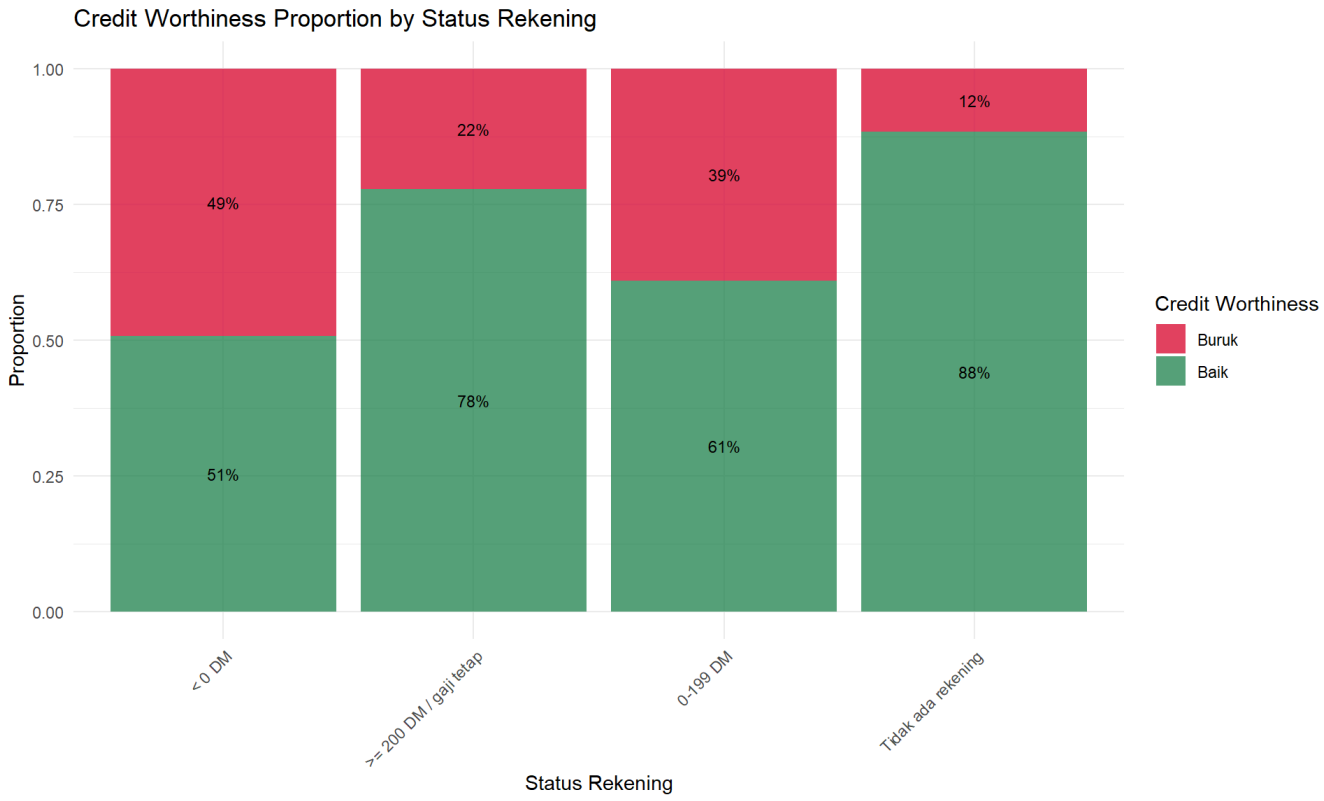
```
# Calculate target distribution
target_dist <- table(data$Kelayakan.Kredit)
target_prop <- prop.table(target_dist) * 100

# Visualize target distribution
plt_target <- ggplot(data, aes(x = Kelayakan.Kredit, fill = Kelayakan.Kredit)) +
    geom_bar(alpha = 0.8) +
    geom_text(stat = "count",
              aes(label = paste0(scales::percent(after_stat(count)/sum(after_stat(co
        unt))),
                                 "\n(n=", after_stat(count), ")")),
              vjust = -0.5, size = 4) +
    scale_fill_manual(values = c("Baik" = "#2E8B57", "Buruk" = "#DC143C")) +
    labs(title = "Distribution of Credit Worthiness",
         subtitle = paste("Good:", round(target_prop[2], 1), "% | Bad:", round(targe
         t_prop[1], 1), "%"),
         x = "Credit Worthiness",
         y = "Count") +
    theme(legend.position = "none")

print(plt_target)
```

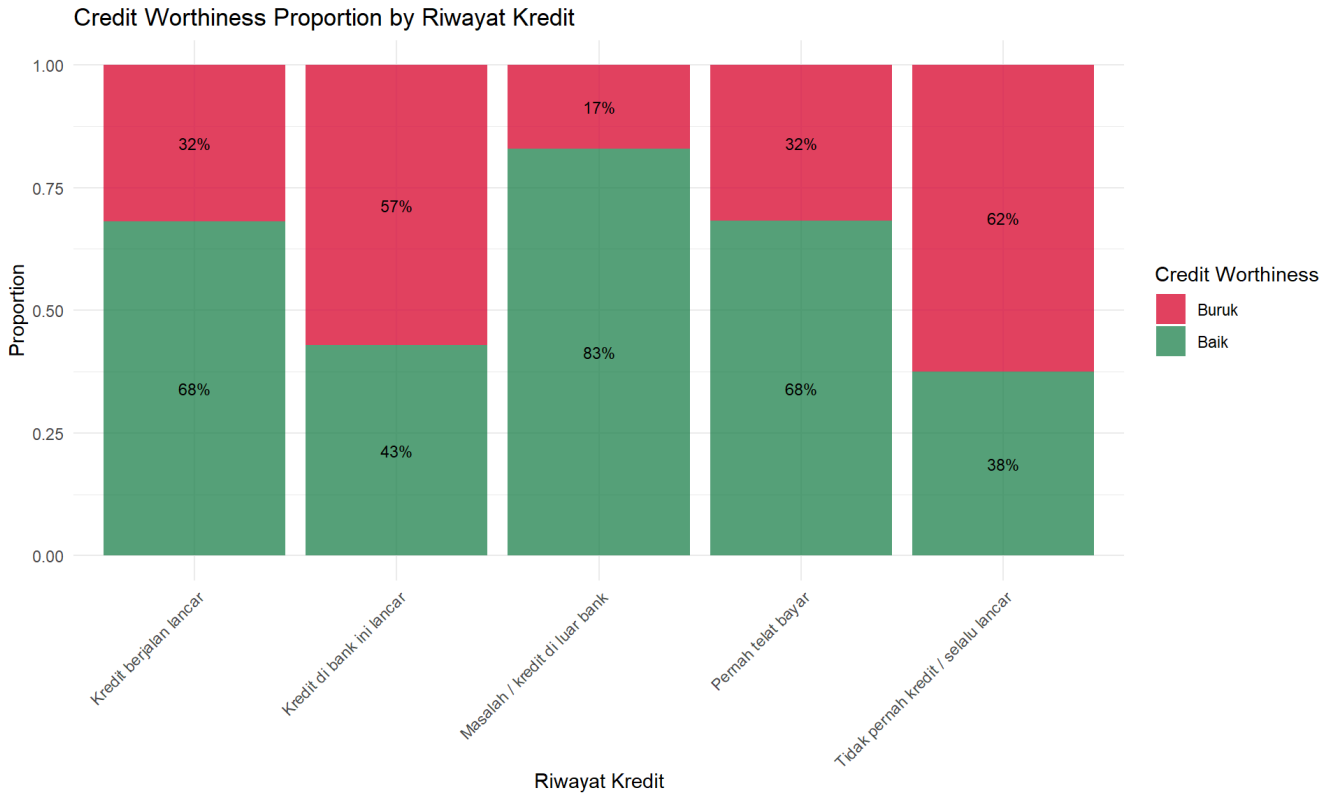


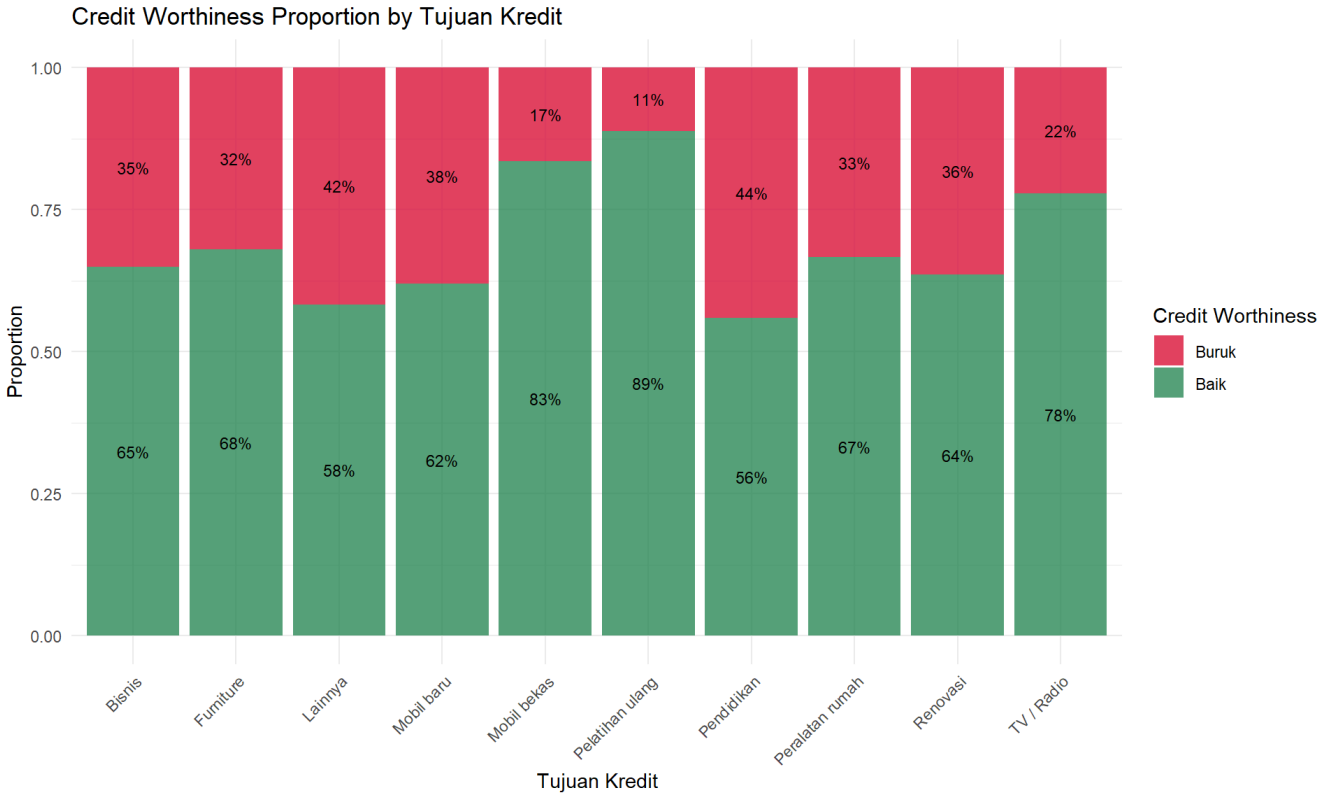
4.2 Categorical Variables Analysis

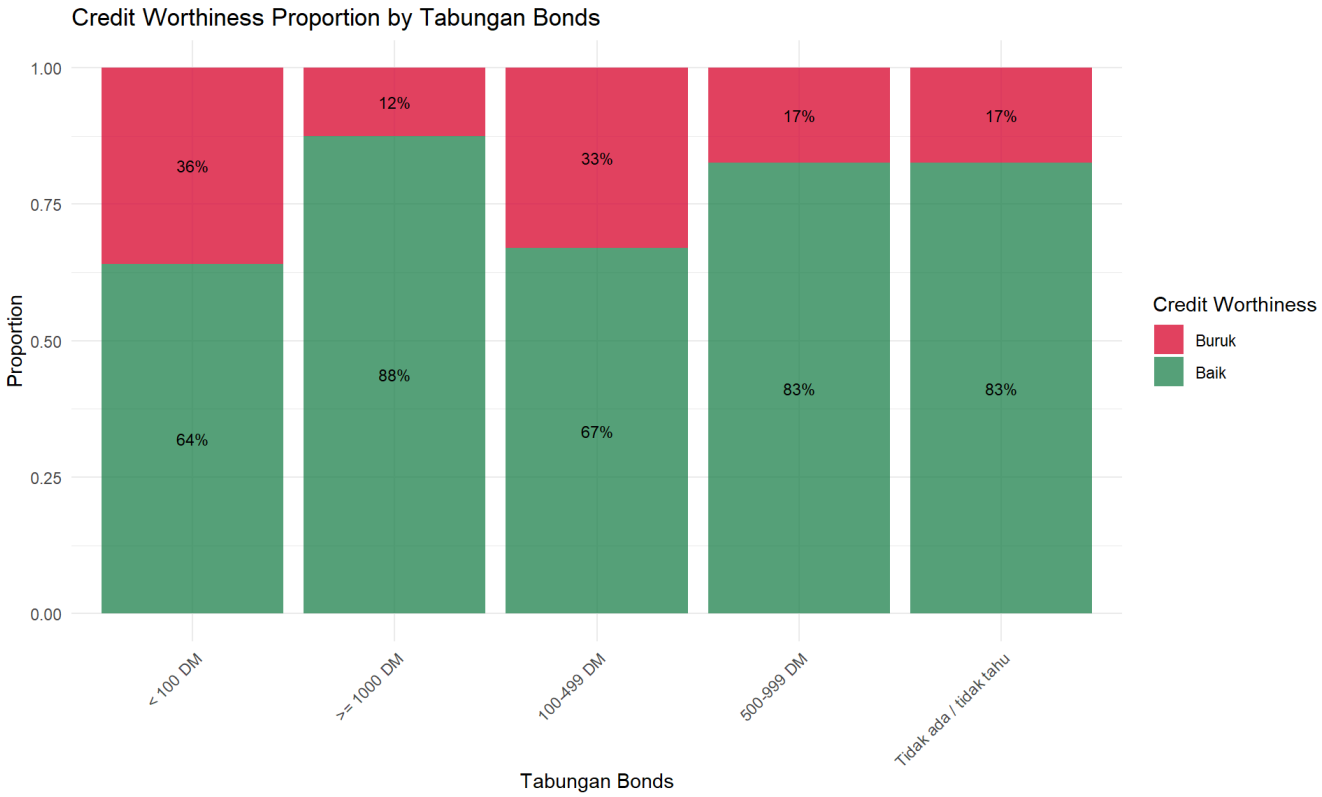
Hide

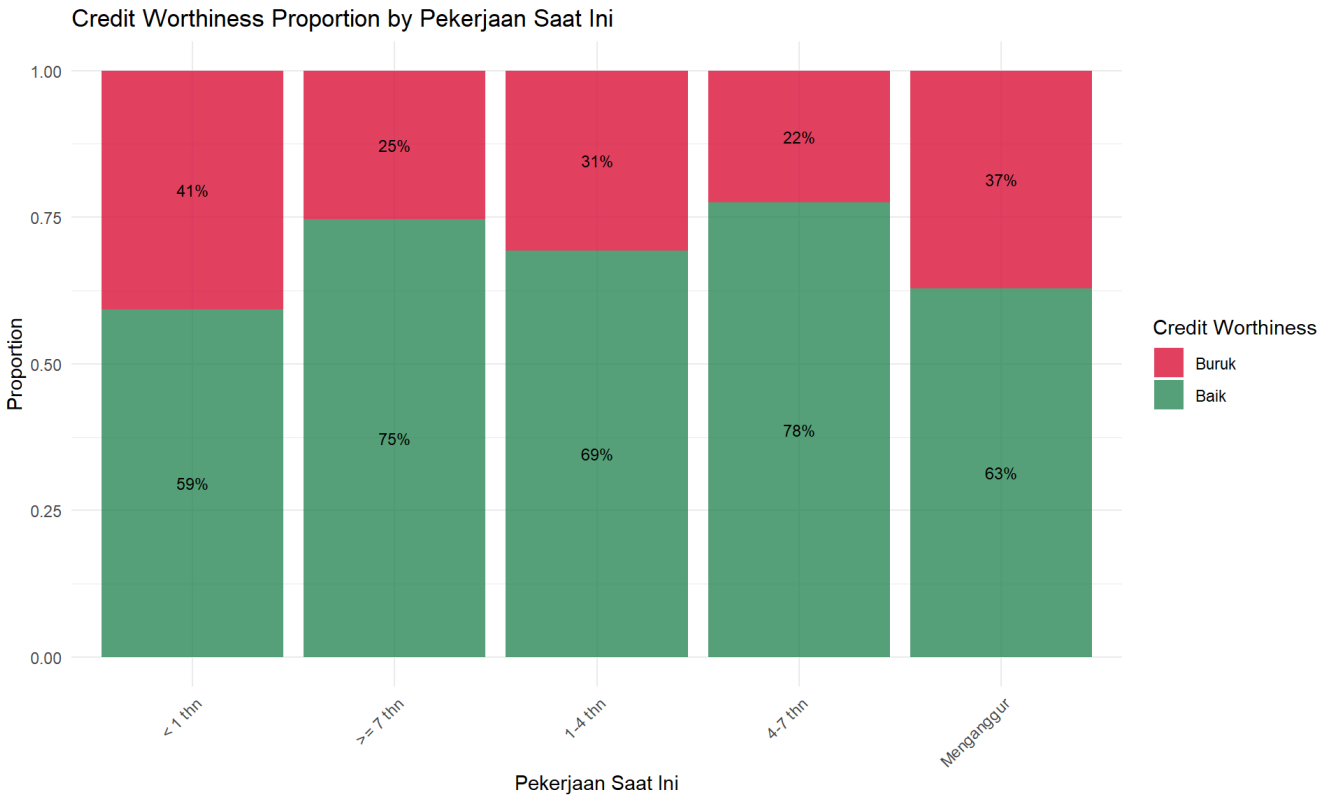
```
# Function to create proportion plots for categorical variables
plot_categorical_variable <- function(data, var_name) {
    prop_df <- data %>%
        group_by(!!sym(var_name), Kelayakan.Kredit) %>%
        summarise(count = n(), .groups = "drop") %>%
        group_by(!!sym(var_name)) %>%
        mutate(
            prop = count / sum(count),
            total = sum(count)
        )

    ggplot(prop_df, aes(x = !!sym(var_name), y = prop, fill = Kelayakan.Kredit)) +
        geom_bar(stat = "identity", position = "fill", alpha = 0.8) +
        geom_text(aes(label = paste0(round(prop*100), "%")),
                  position = position_fill(vjust = 0.5), size = 3) +
        scale_fill_manual(values = c("Baik" = "#2E8B57", "Buruk" = "#DC143C")) +
        labs(title = paste("Credit Worthiness Proportion by", gsub("\\.", " ", var_n
         ame)),
             x = gsub("\\.", " ", var_name),
             y = "Proportion",
             fill = "Credit Worthiness") +
        theme(axis.text.x = element_text(angle = 45, hjust = 1))
}

# Important categorical variables to analyze
important_cat_vars <- c("Status.Rekening", "Riwayat.Kredit", "Tujuan.Kredit",
                        "Tabungan.Bonds", "Pekerjaan.Saat.Ini",
                        "Status.Pribadi.dan.Jenis.Kelamin", "Properti")

# Generate plots for categorical variables
cat_plots <- map(important_cat_vars, ~plot_categorical_variable(data, .x))
names(cat_plots) <- important_cat_vars

# Display plots
walk(cat_plots, print)
```


Credit Worthiness Proportion by Status Rekening
1.00
0.75
0.50
0.25
0.00
Proportion
49%
51%
22%
78%
39%
61%
12%
88%
< 0 DM
>= 200 DM / gaji tetap
0-199 DM
Tidak ada rekening
Status Rekening
Credit Worthiness
Buruk
Baik

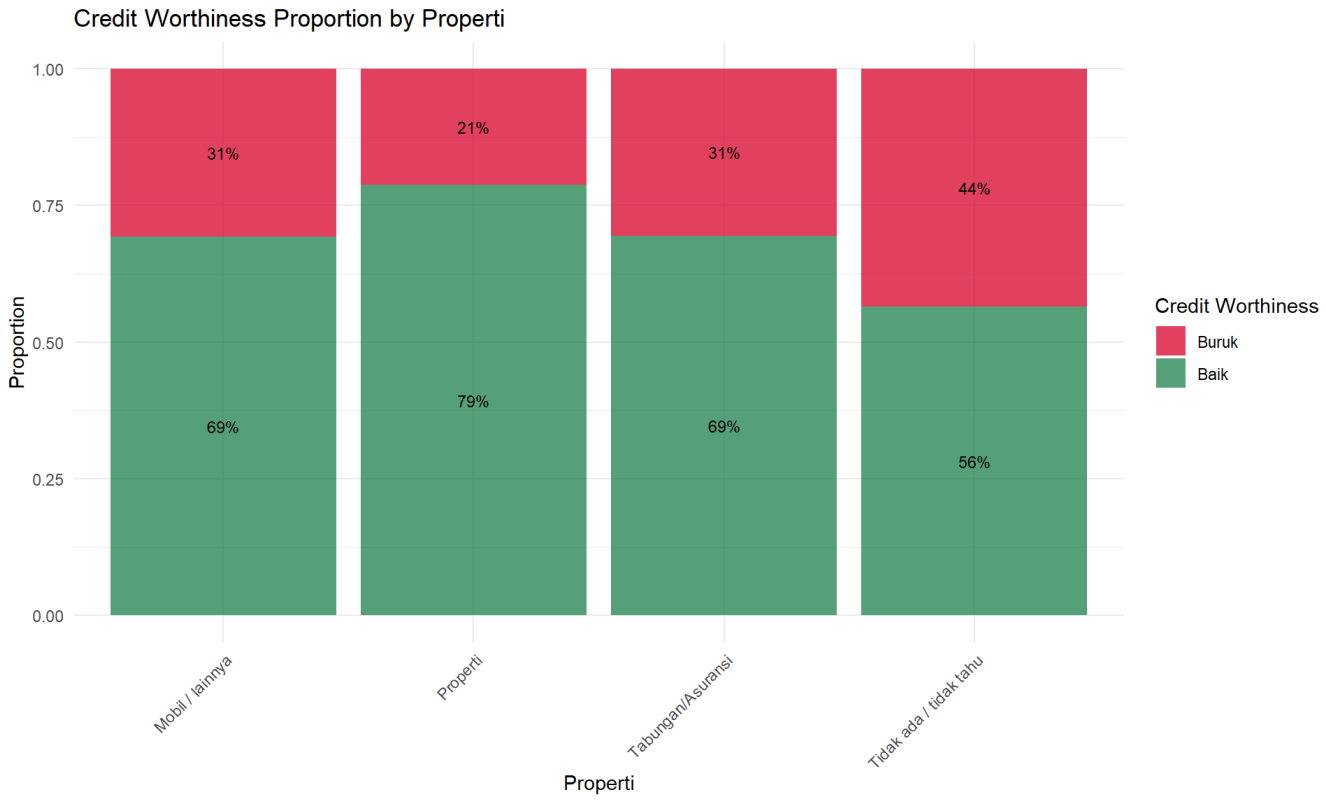

Credit Worthiness Proportion by Riwayat Kredit
1.00
0.75
0.50
0.25
0.00
Proportion
32%
68%
57%
43%
17%
83%
32%
68%
62%
38%
Kredit berjalan lancar
Kredit di bank ini lancar
Masalah / kredit di luar bank
Pernah telat bayar
Tidak pernah kredit / selalu lancar
Riwayat Kredit
Credit Worthiness
Buruk
Baik


Credit Worthiness Proportion by Tujuan Kredit
Proportion
1.00
0.75
0.50
0.25
0.00
35%
32%
42%
38%
17%
11%
44%
33%
36%
22%
65%
68%
58%
62%
83%
89%
56%
67%
64%
78%
Bisnis
Furniture
Lainnya
Mobil baru
Mobil bekas
Pelatihan ulang
Pendidikan
Peralatan rumah
Renovasi
TV / Radio
Tujuan Kredit
Credit Worthiness
Buruk
Baik


Credit Worthiness Proportion by Tabungan Bonds
Proportion
1.00
0.75
0.50
0.25
0.00
36%
12%
33%
17%
17%
64%
88%
67%
83%
83%
< 100 DM
>= 1000 DM
100-499 DM
500-999 DM
Tidak ada / tidak tahu
Tabungan Bonds
Credit Worthiness
Buruk
Baik


Credit Worthiness Proportion by Pekerjaan Saat Ini
Proportion
1.00
0.75
0.50
0.25
0.00
41%
25%
31%
22%
37%
59%
75%
69%
78%
63%
< 1 thn
>= 7 thn
1-4 thn
4-7 thn
Menganggur
Pekerjaan Saat Ini
Credit Worthiness
Buruk
Baik

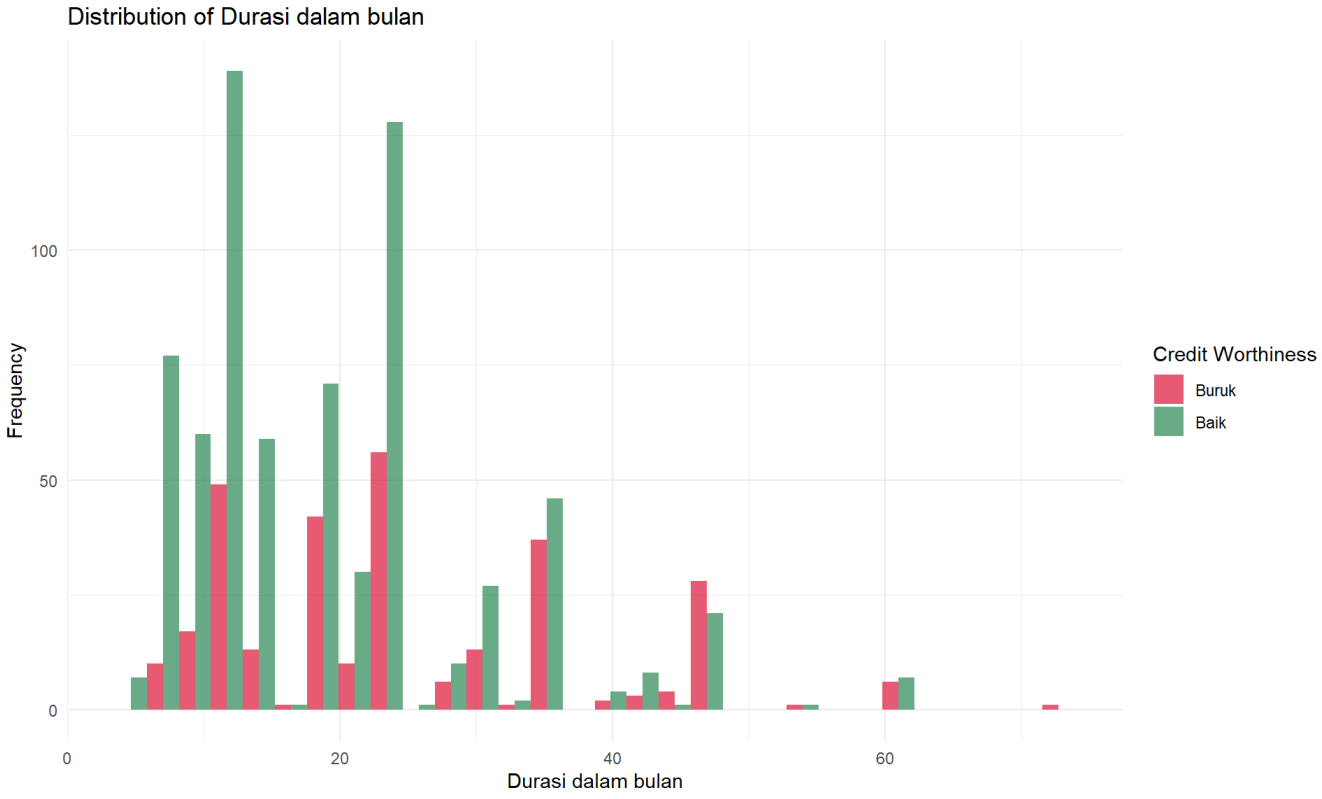

Credit Worthiness Proportion by Status Pribadi dan Jenis Kelamin
Proportion
1.00
0.75
0.50
0.25
0.00
40%
27%
27%
35%
60%
73%
73%
65%
Pria cerai/pisah
Pria lajang
Pria menikah/duda
Wanita menikah/cerai
Status Pribadi dan Jenis Kelamin
Credit Worthiness
Buruk
Baik

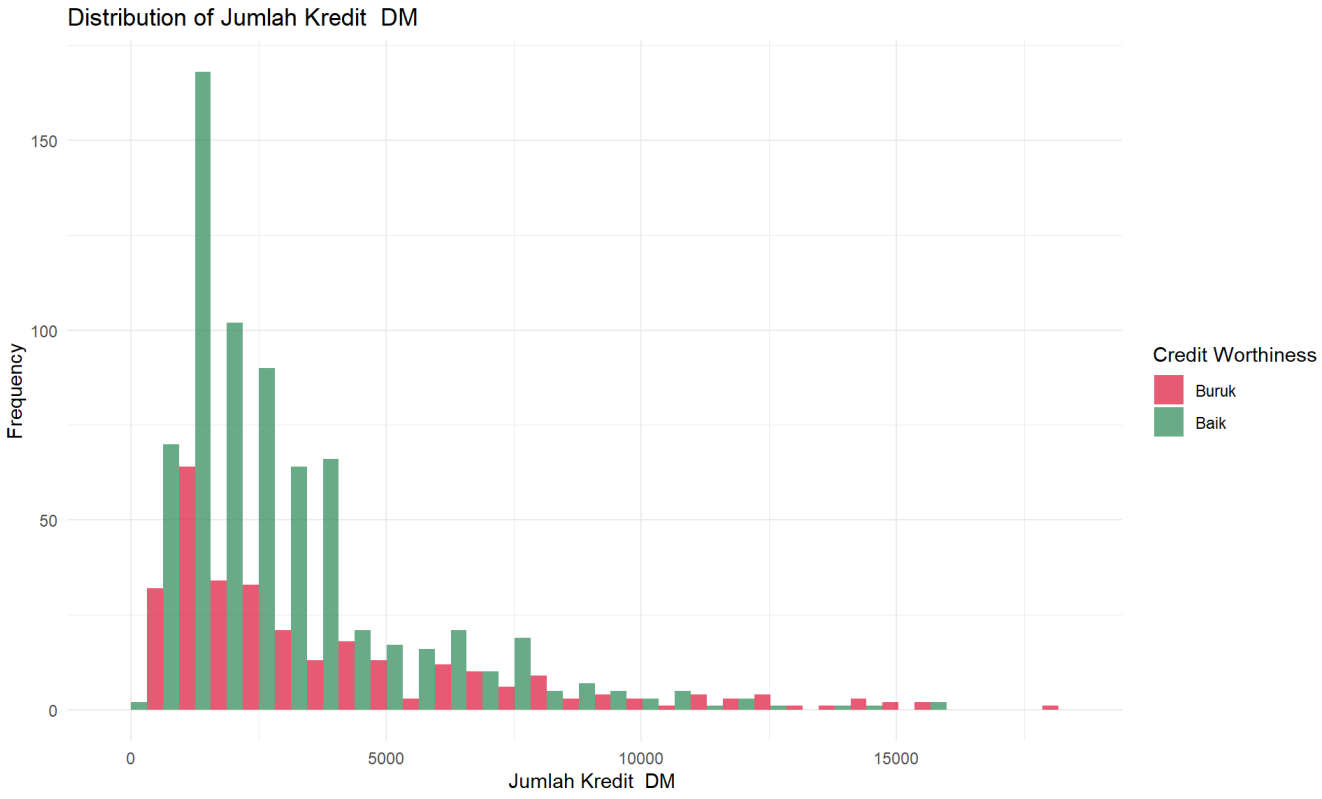


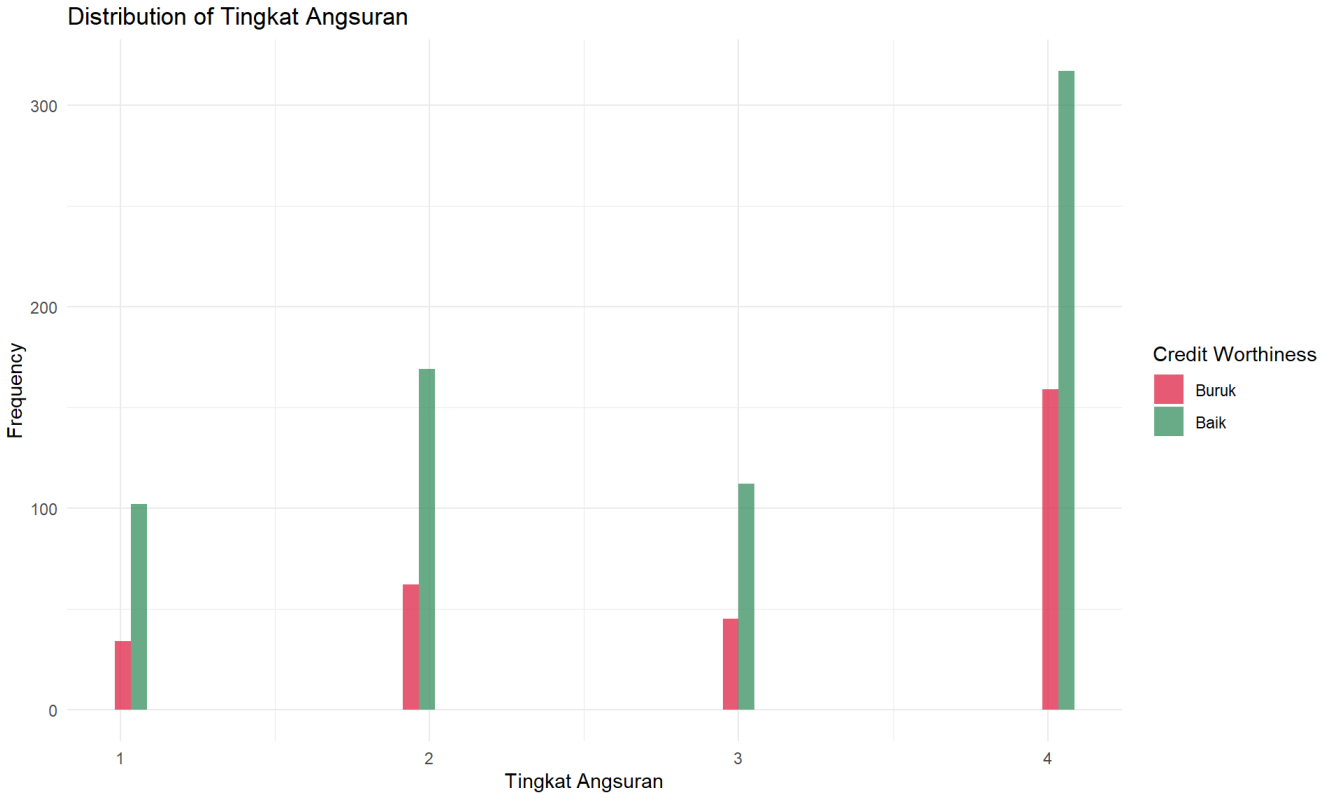
4.3 Numerical Variables Analysis

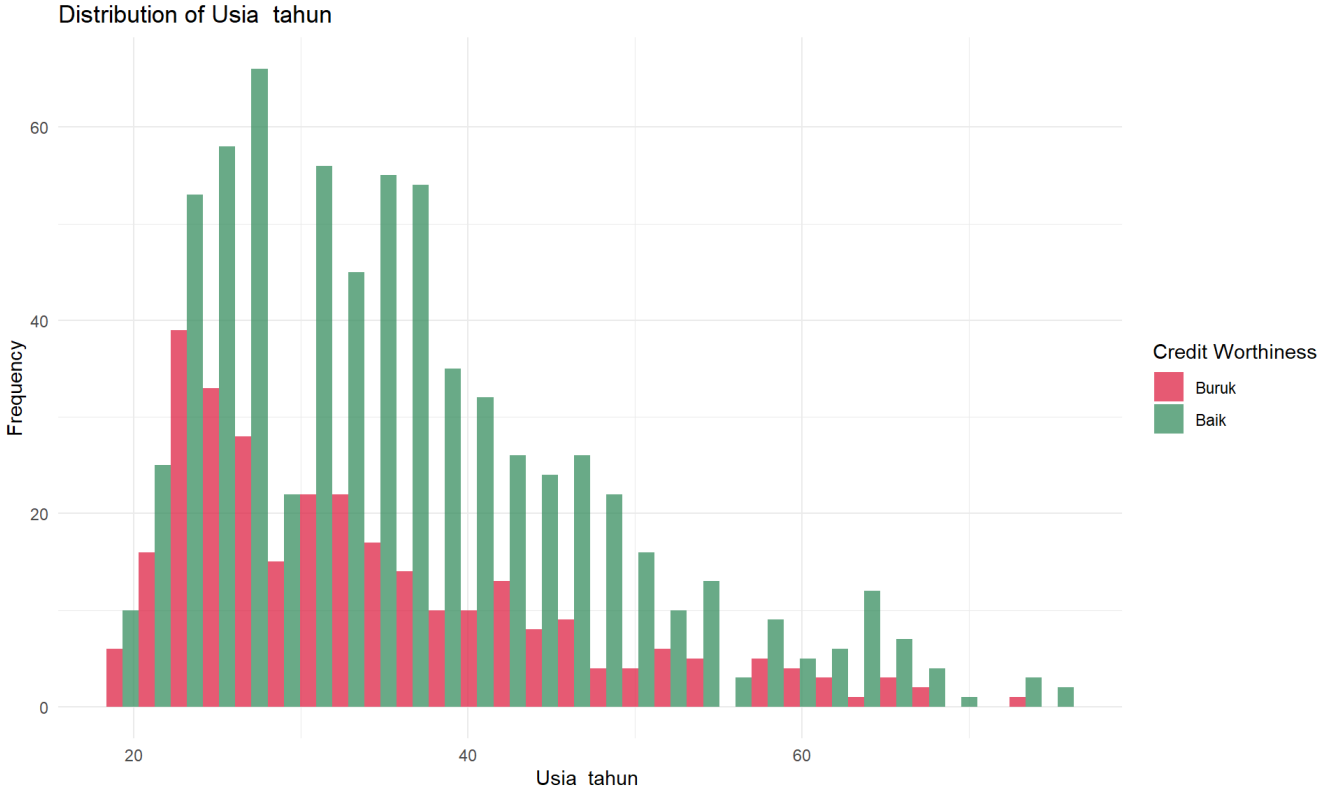
Hide

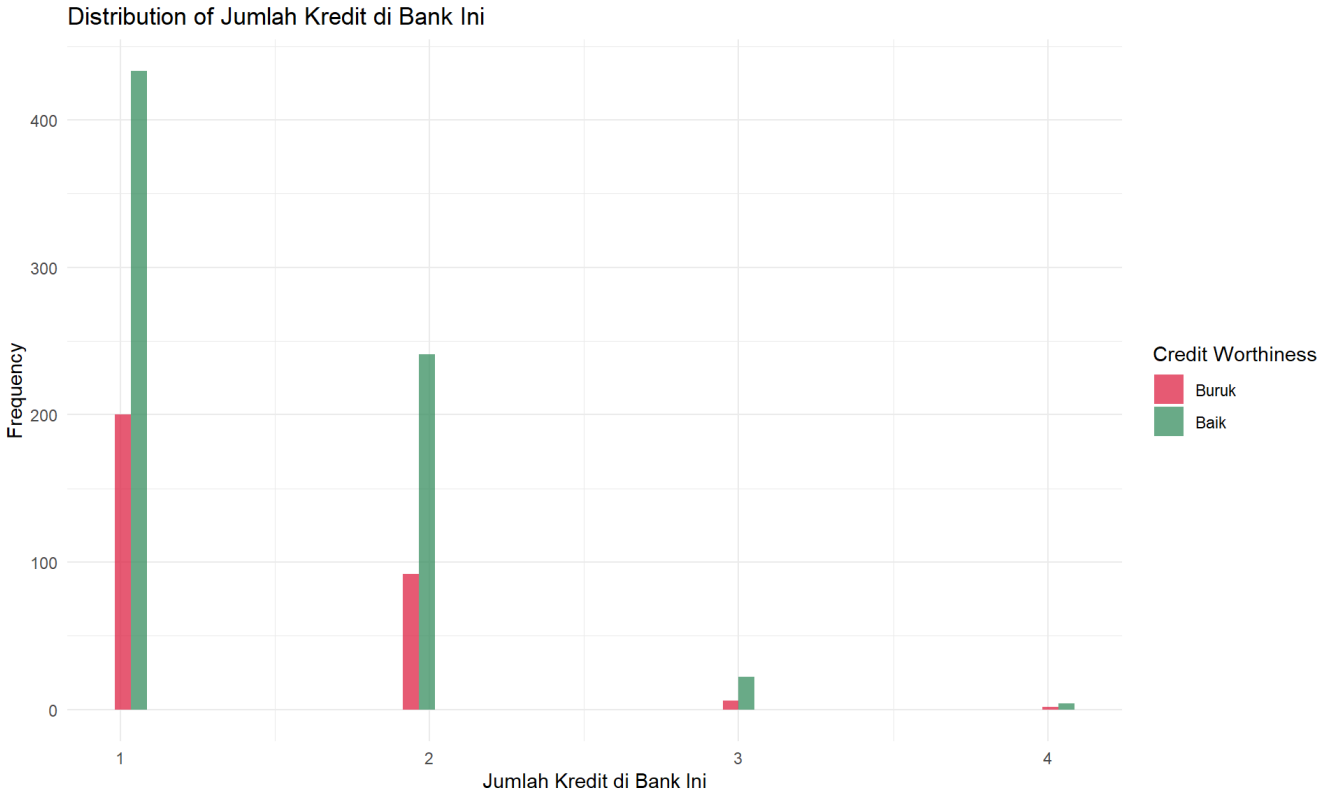
```
# Identify numerical variables
numerical_vars <- c("Durasi.dalam.bulan", "Jumlah.Kredit..DM.", "Tingkat.Angsura
        n....",
                    "Usia..tahun.", "Jumlah.Kredit.di.Bank.Ini", "Jumlah.Tanggungan")

# Create histograms for numerical variables
create_histogram <- function(data, var_name) {
    ggplot(data, aes_string(x = var_name, fill = "Kelayakan.Kredit")) +
        geom_histogram(position = "dodge", alpha = 0.7, bins = 30) +
        scale_fill_manual(values = c("Baik" = "#2E8B57", "Buruk" = "#DC143C")) +
        labs(title = paste("Distribution of", gsub("\\.", " ", var_name)),
             x = gsub("\\.", " ", var_name),
             y = "Frequency",
             fill = "Credit Worthiness") +
        theme_minimal()
}

# Generate histograms
num_histograms <- map(numerical_vars, ~create_histogram(data, .x))
names(num_histograms) <- numerical_vars

# Display histograms
walk(num_histograms, print)
```

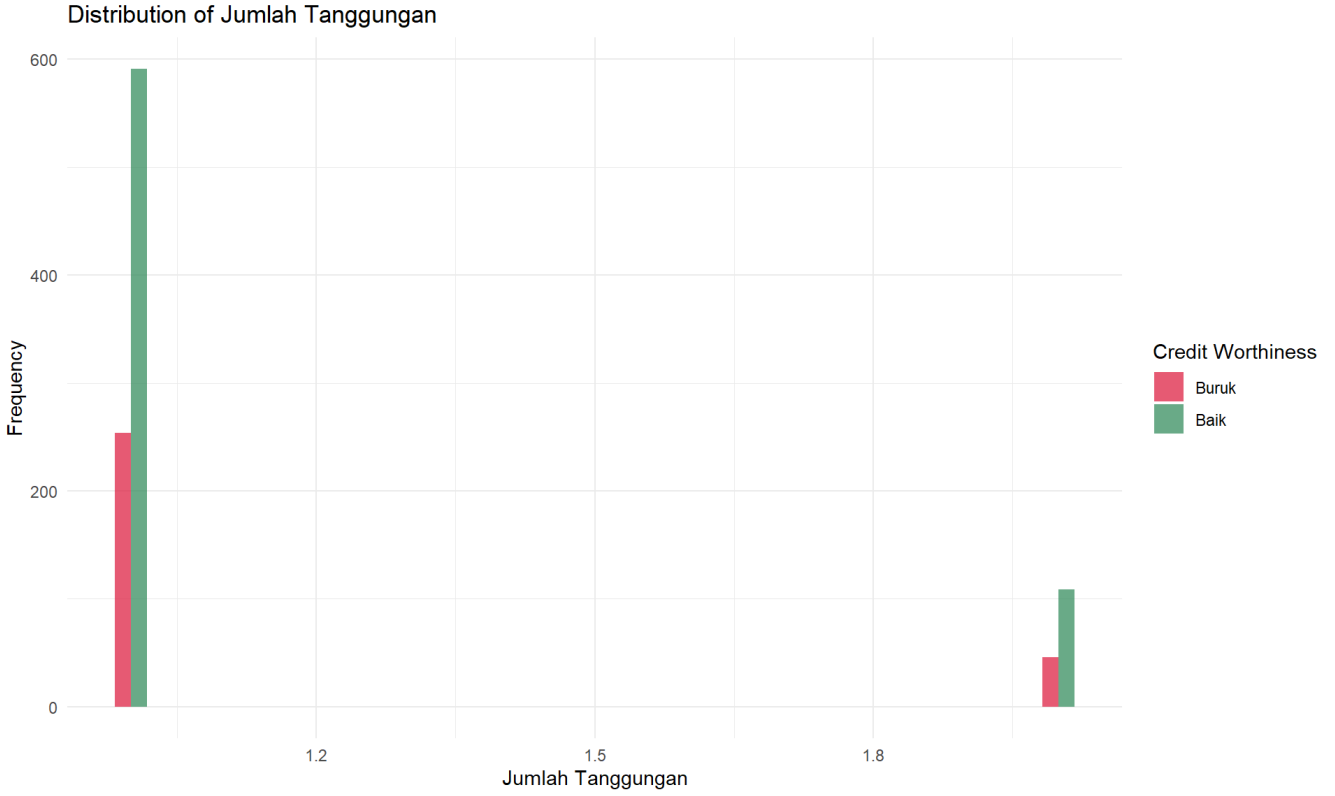

Distribution of Durasi dalam bulan
Frequency
100
50
0
0
20
40
60
Durasi dalam bulan
Credit Worthiness
Buruk
Baik


Distribution of Jumlah Kredit DM
Frequency
150
100
50
0
0
5000
10000
15000
Jumlah Kredit DM
Credit Worthiness
Buruk
Baik


Distribution of Tingkat Angsuran
Frequency
300
200
100
0
1
2
3
4
Tingkat Angsuran
Credit Worthiness
Buruk
Baik

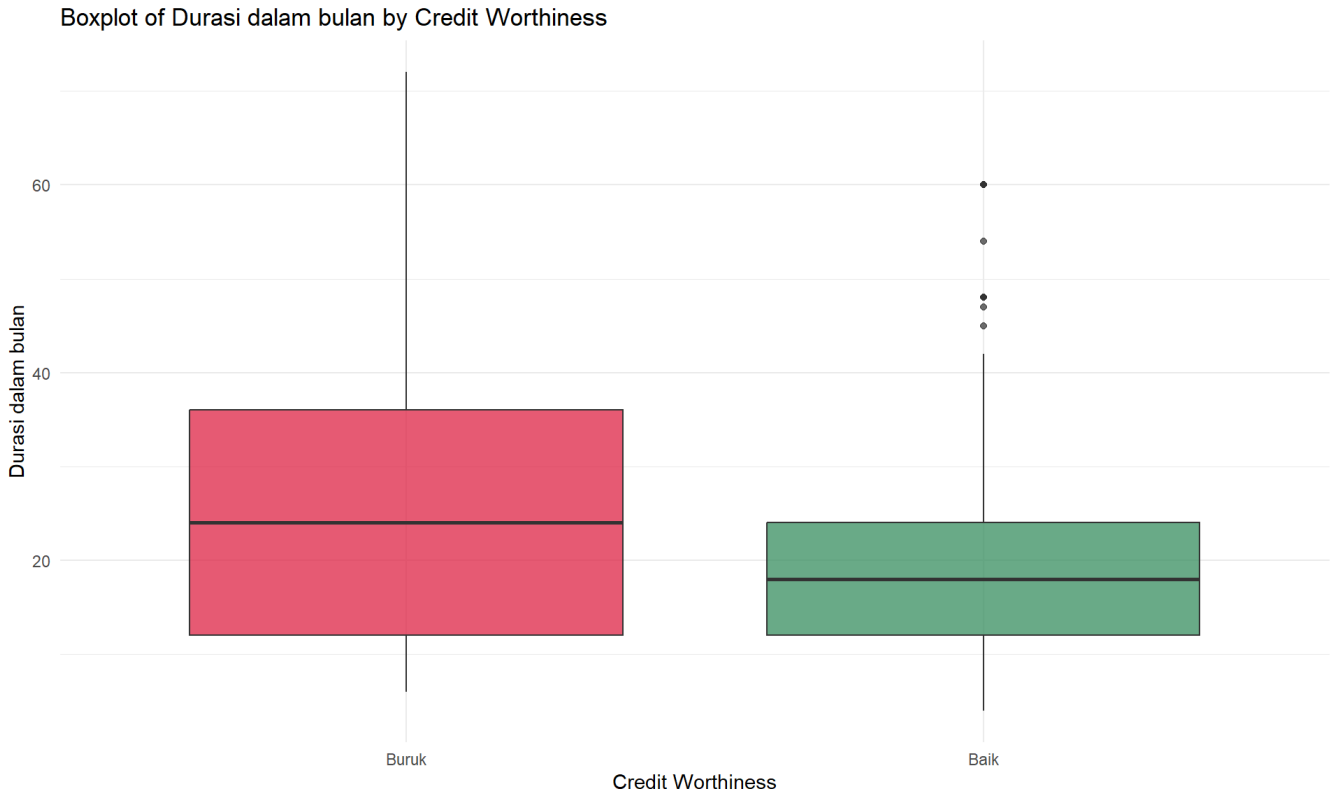

Distribution of Usia tahun
Frequency
60
40
20
0
20
40
60
Usia tahun
Credit Worthiness
Buruk
Baik

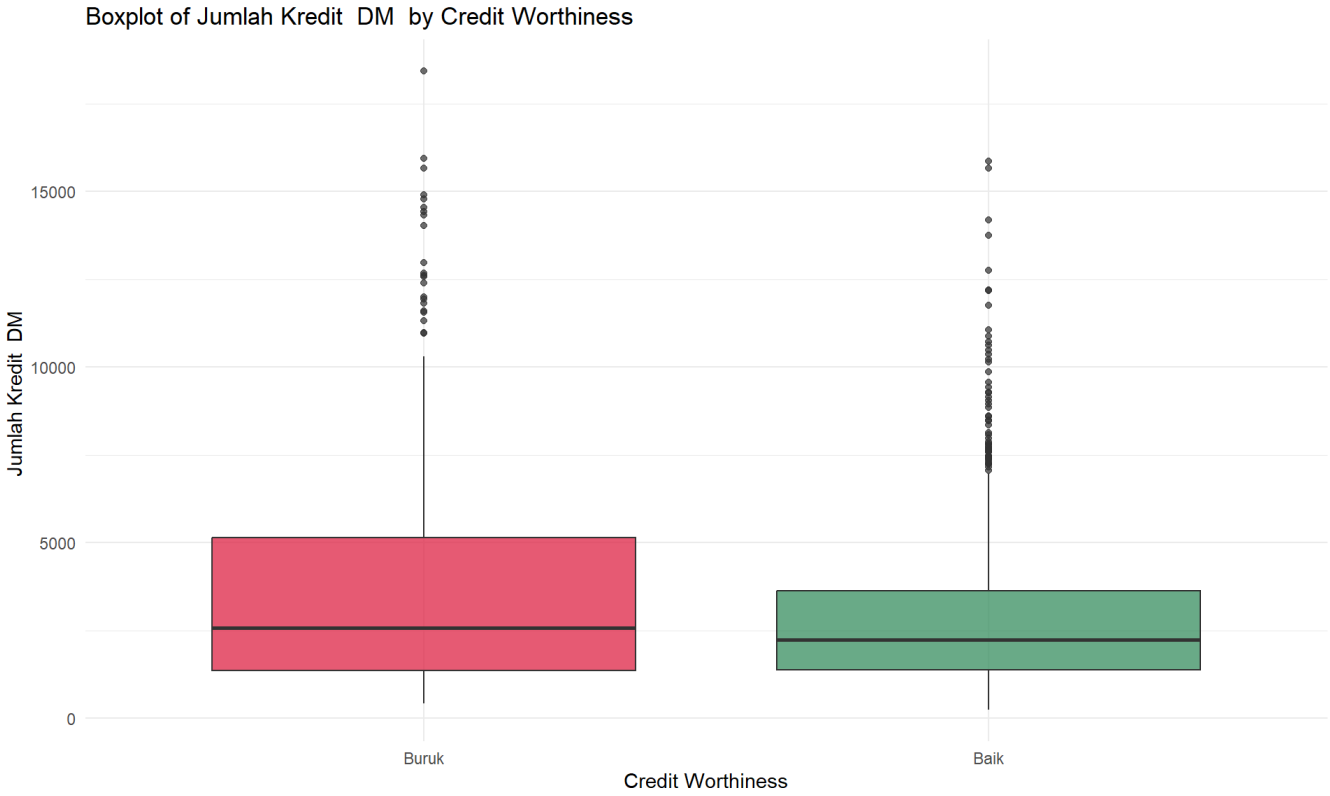

Distribution of Jumlah Kredit di Bank Ini
Frequency
400
300
200
100
0
1
2
3
4
Jumlah Kredit di Bank Ini
Credit Worthiness
Buruk
Baik

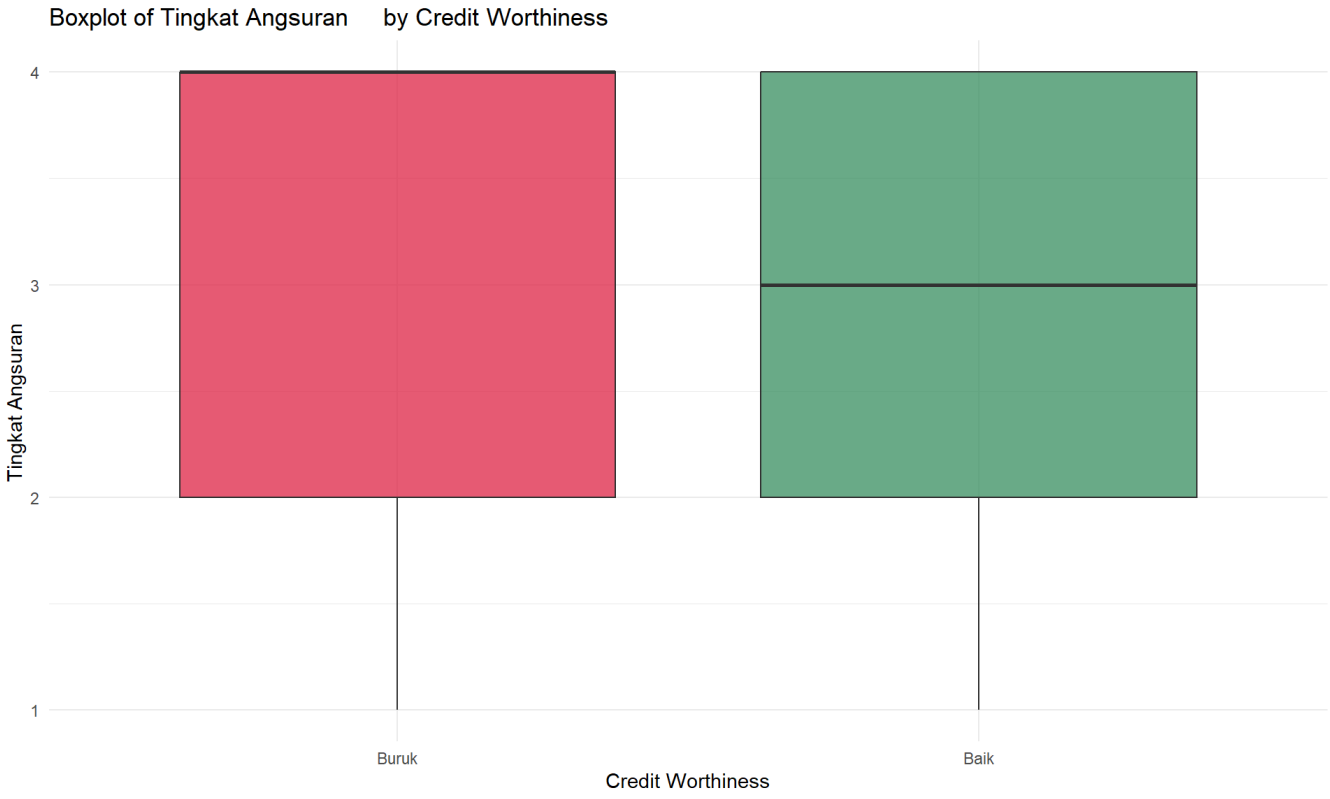

Distribution of Jumlah Tanggungan
Frequency
600
400
200
0
1.2
1.5
1.8
Jumlah Tanggungan
Credit Worthiness
Buruk
Baik

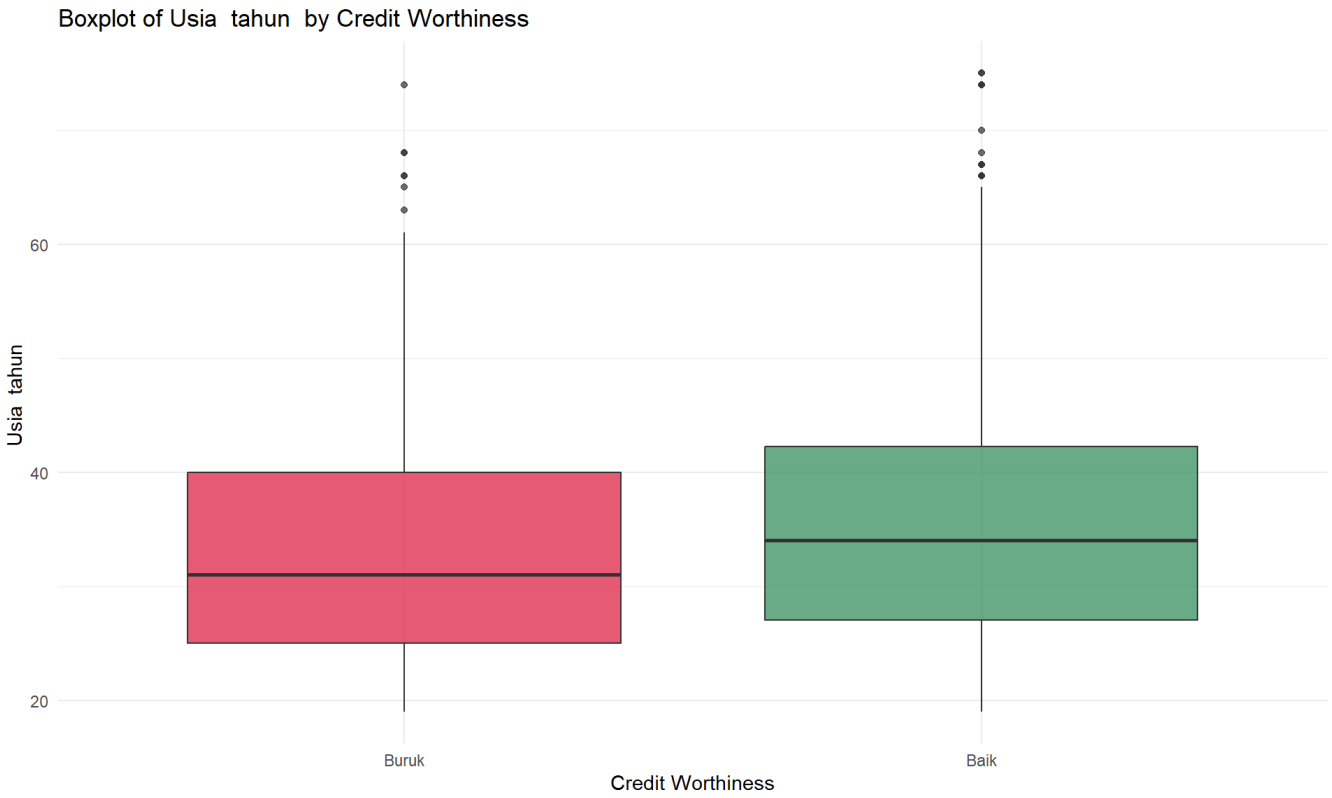
Hide

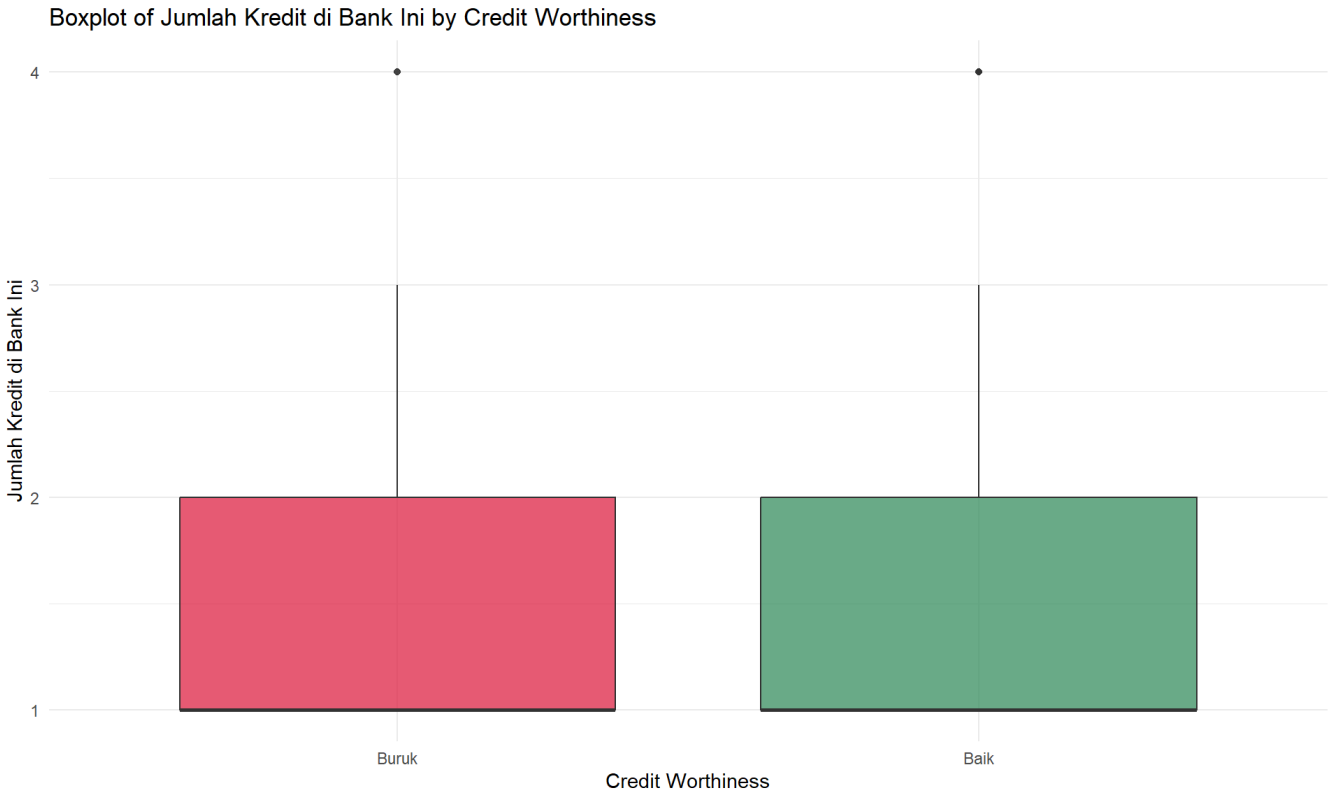
```
# Create boxplots for numerical variables
create_boxplot <- function(data, var_name) {
    ggplot(data, aes_string(x = "Kelayakan.Kredit", y = var_name, fill = "Kelayakan.
        Kredit")) +
        geom_boxplot(alpha = 0.7) +
        scale_fill_manual(values = c("Baik" = "#2E8B57", "Buruk" = "#DC143C")) +
        labs(title = paste("Boxplot of", gsub("\\.", " ", var_name), "by Credit Wort
        hiness"),
            x = "Credit Worthiness",
            y = gsub("\\.", " ", var_name)) +
        theme(legend.position = "none")
}

# Generate boxplots
num_boxplots <- map(numerical_vars, ~create_boxplot(data, .x))
names(num_boxplots) <- numerical_vars

# Display boxplots
walk(num_boxplots, print)
```

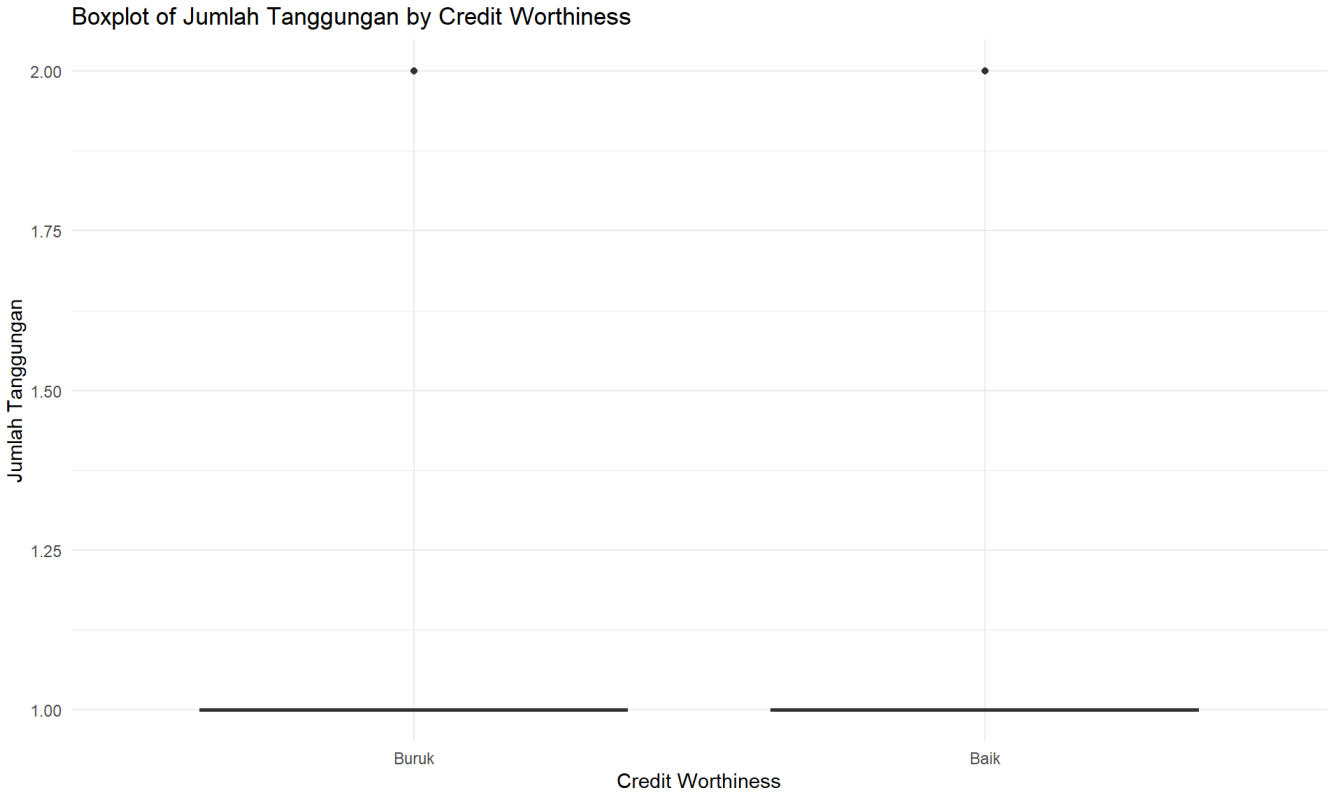



Boxplot of Jumlah Kredit DM by Credit Worthiness
Jumlah Kredit DM
15000
10000
5000
0
Buruk
Baik
Credit Worthiness


Boxplot of Tingkat Angsuran by Credit Worthiness
Tingkat Angsuran
4
3
2
1
Buruk
Baik
Credit Worthiness


Boxplot of Usia tahun by Credit Worthiness
Usia tahun
60
40
20
Buruk
Baik
Credit Worthiness

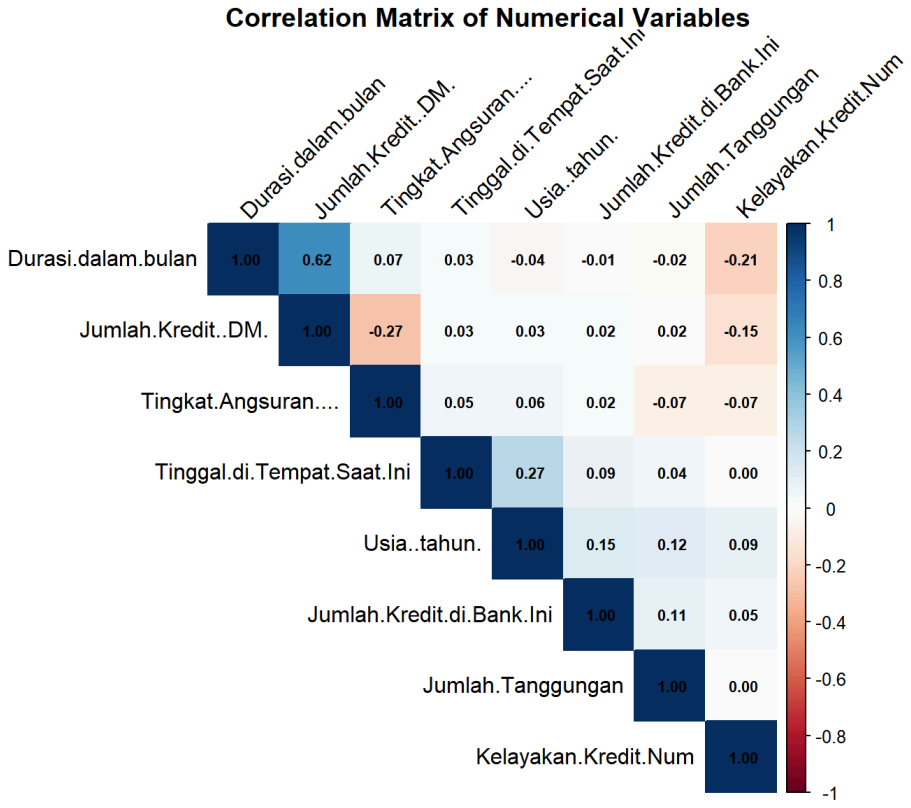

Boxplot of Jumlah Kredit di Bank Ini by Credit Worthiness
Jumlah Kredit di Bank Ini
4
3
2
1
Buruk
Baik
Credit Worthiness



4.4 Correlation Analysis

Hide

```
# Create numeric version of target for correlation
data$Kelayakan.Kredit.Num <- ifelse(data$Kelayakan.Kredit == "Baik", 1, 0)

# Select numerical variables for correlation analysis
num_data <- data %>% select_if(is.numeric)

# Calculate correlation matrix
corr_matrix <- cor(num_data, use = "complete.obs")

# Visualize correlation matrix
corrplot(corr_matrix,
         method = "color",
         type = "upper",
         tl.col = "black",
         tl.srt = 45,
         addCoef.col = "black",
         number.cex = 0.7,
         title = "Correlation Matrix of Numerical Variables",
         mar = c(0,0,1,0))
```



5. Data Preprocessing and Model Preparation

Hide

```
# Prepare data for modeling
model_data <- data %>% select(-Kelayakan.Kredit.Num)

# Split data into training and testing sets
set.seed(123)
train_index <- createDataPartition(model_data$Kelayakan.Kredit, p = 0.7, list = FALS
        E)
train_data <- model_data[train_index, ]
test_data <- model_data[-train_index, ]

cat("Training set size:", nrow(train_data), "\n")
```

```
## Training set size: 700
```

Hide

```
cat("Test set size:", nrow(test_data), "\n")
```

```
## Test set size: 300
```

Hide

```
cat("Training set class distribution:\n")
```

```
## Training set class distribution:
```

Hide

```
print(table(train_data$Kelayakan.Kredit))
```

```
##
## Buruk  Baik
##   210   490
```

Hide

```
# Handle class imbalance using ROSE
train_balanced <- ROSE(Kelayakan.Kredit ~ ., data = train_data, seed = 123)$data

cat("Balanced training set class distribution:\n")
```

```
## Balanced training set class distribution:
```

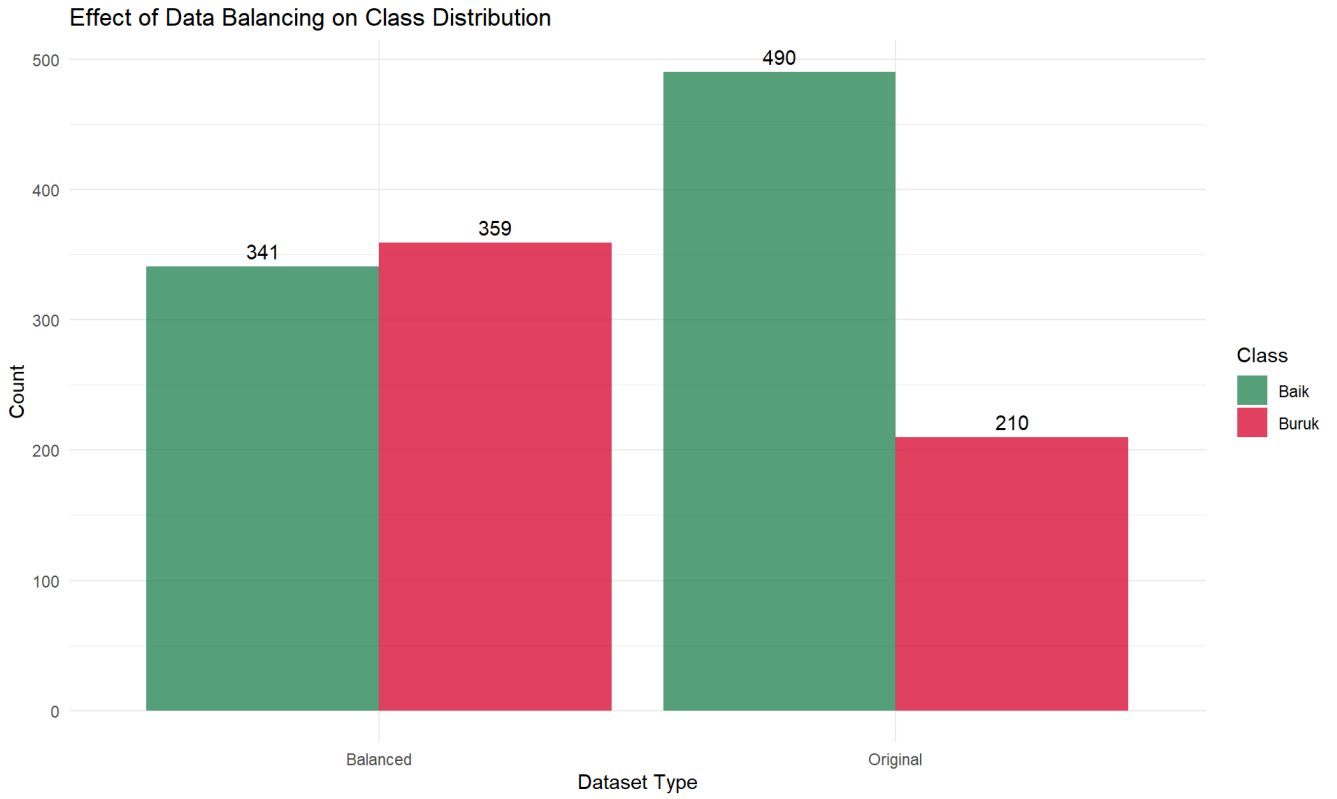
Hide

```
print(table(train_balanced$Kelayakan.Kredit))
```

```
##
##  Baik Buruk
##   359   341
```

Hide

```
# Visualize the effect of balancing
balance_comparison <- data.frame(
    Dataset = rep(c("Original", "Balanced"), each = 2),
    Class = rep(c("Buruk", "Baik"), 2),
    Count = c(table(train_data$Kelayakan.Kredit), table(train_balanced$Kelayakan.Kre
        dit))
)

ggplot(balance_comparison, aes(x = Dataset, y = Count, fill = Class)) +
    geom_bar(stat = "identity", position = "dodge", alpha = 0.8) +
    scale_fill_manual(values = c("Baik" = "#2E8B57", "Buruk" = "#DC143C")) +
    labs(title = "Effect of Data Balancing on Class Distribution",
        x = "Dataset Type", y = "Count") +
    geom_text(aes(label = Count), position = position_dodge(width = 0.9), vjust = -
        0.5)
```



6. CART Model Development

6.1 Basic CART Model

Hide

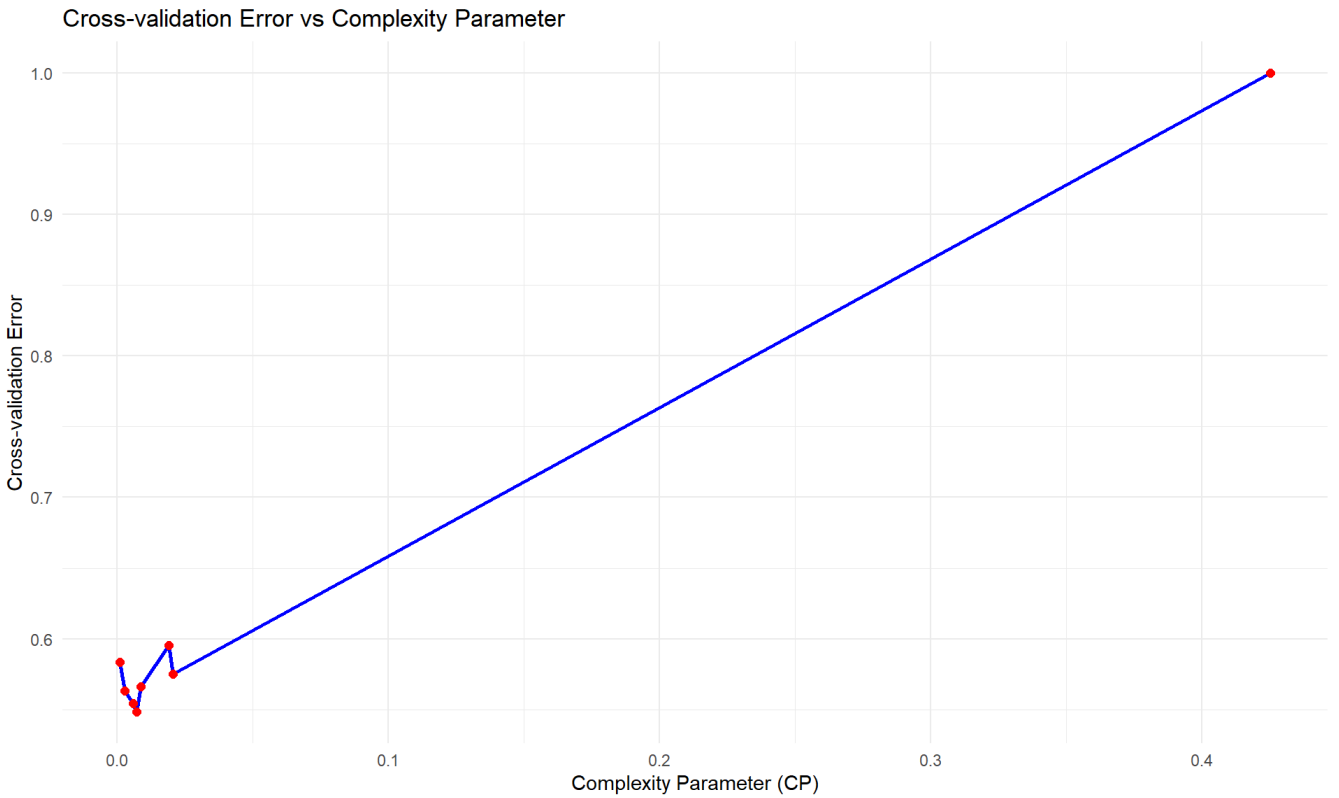
```
# Fit CART model
cart_model <- rpart(
    Kelayakan.Kredit ~ .,
    data = train_balanced,
    method = "class",
    control = rpart.control(
        cp = 0.01,
        minbucket = 5,
        minsplit = 10
    )
)

# Print model summary
printcp(cart_model)
```

```
##
## Classification tree:
## rpart(formula = Kelayakan.Kredit ~ ., data = train_balanced,
##     method = "class", control = rpart.control(cp = 0.01, minbucket = 5,
##         minsplit = 10))
##
## Variables actually used in tree construction:
## [1] Durasi.dalam.bulan Pekerjaan          Pekerjaan.Saat.Ini Properti
## [5] Riwayat.Kredit     Status.Rekening    Tabungan.Bonds     Tujuan.Kredit
##
## Root node error: 341/700 = 0.48714
##
## n= 700
##
##         CP nsplit rel error  xerror     xstd
## 1 0.425220      0   1.00000 1.00000 0.038781
## 2 0.020528      1   0.57478 0.57478 0.034837
## 3 0.019062      5   0.48680 0.60997 0.035458
## 4 0.010000      8   0.42815 0.60411 0.035358
```

6.2 CART Model Pruning

Hide

```
# Build full tree for pruning analysis
cart_model_full <- rpart(
    Kelayakan.Kredit ~ .,
    data = train_balanced,
    method = "class",
    control = rpart.control(
        cp = 0.001,
        minbucket = 5,
        minsplit = 10
    )
)

# Plot cross-validation error vs complexity parameter
cp_table <- as.data.frame(cart_model_full$cptable)

ggplot(cp_table, aes(x = CP, y = xerror)) +
    geom_line(color = "blue", size = 1) +
    geom_point(color = "red", size = 2) +
    labs(title = "Cross-validation Error vs Complexity Parameter",
         x = "Complexity Parameter (CP)",
         y = "Cross-validation Error") +
    theme_minimal()
```



Hide

```
# Find optimal CP using 1-SE rule
min_error <- min(cart_model_full$cptable[,"xerror"])
se_min_error <- cart_model_full$cptable[which.min(cart_model_full$cptable[,"xerro
        r"]), "xstd"]
threshold <- min_error + se_min_error
optimal_cp <- cart_model_full$cptable[cart_model_full$cptable[,"xerror"] <= threshol
        d, "CP"][1]

cat("Optimal CP (1-SE rule):", optimal_cp, "\n")
```

```
## Optimal CP (1-SE rule): 0.02052786
```

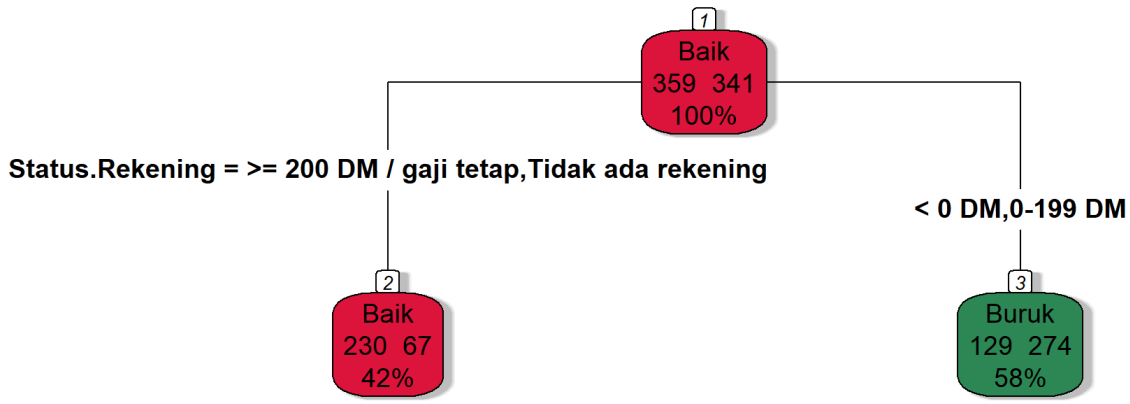
Hide

```
# Prune the tree
cart_model_pruned <- prune(cart_model_full, cp = optimal_cp)
```

6.3 CART Model Visualization

Hide

```
# Visualize pruned decision tree
rpart.plot(
    cart_model_pruned,
    type = 4,
    extra = 101,
    box.palette = c("#DC143C", "#2E8B57"),
    branch.lty = 1,
    shadow.col = "gray",
    nn = TRUE,
    tweak = 1.2,
    fallen.leaves = TRUE,
    uniform = TRUE,
    main = "Pruned CART Decision Tree for Credit Worthiness Prediction"
)
```



Hide

```
# Save high-quality decision tree plot
if(!dir.exists("plots")) dir.create("plots")

pdf("plots/decision_tree_pruned.pdf", width = 12, height = 8)
rpart.plot(
    cart_model_pruned,
    type = 4,
    extra = 101,
    box.palette = c("#DC143C", "#2E8B57"),
    branch.lty = 1,
    shadow.col = "gray",
    nn = TRUE,
    tweak = 1.2,
    fallen.leaves = TRUE,
    uniform = TRUE,
    main = "Pruned CART Decision Tree for Credit Worthiness Prediction"
)
dev.off()
```

```
## png
##   2
```

Hide

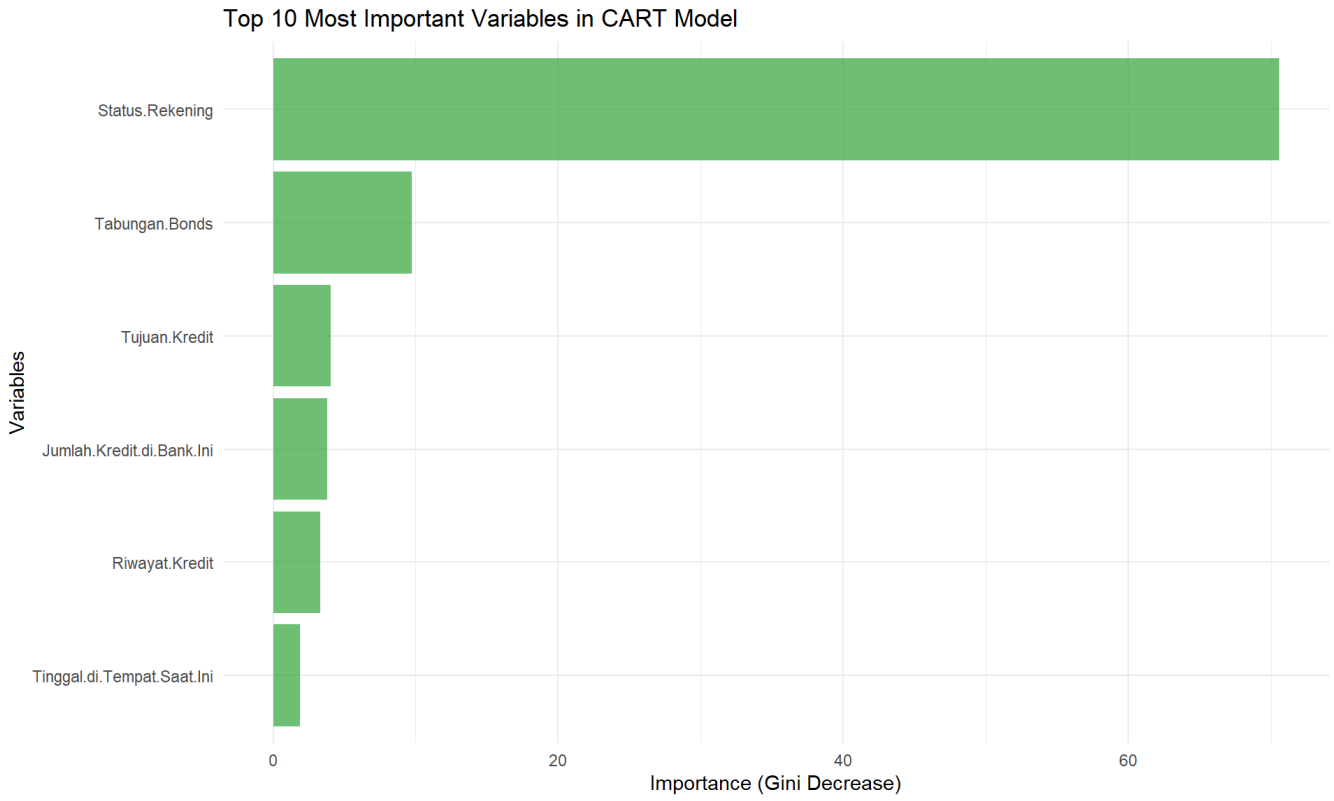
```
cat("Decision tree saved to: plots/decision_tree_pruned.pdf\n")
```

```
## Decision tree saved to: plots/decision_tree_pruned.pdf
```

6.4 CART Variable Importance

Hide

```
# Extract and visualize variable importance
cart_importance <- cart_model_pruned$variable.importance
if(length(cart_importance) > 0) {
    cart_importance_df <- data.frame(
        Variable = names(cart_importance),
        Importance = as.numeric(cart_importance)
    ) %>%
    arrange(desc(Importance)) %>%
    slice_head(n = 10)

    ggplot(cart_importance_df, aes(x = reorder(Variable, Importance), y = Importanc
         e)) +
        geom_bar(stat = "identity", fill = "#4CAF50", alpha = 0.8) +
        coord_flip() +
        labs(title = "Top 10 Most Important Variables in CART Model",
             x = "Variables", y = "Importance (Gini Decrease)") +
        theme_minimal()
} else {
    cat("No variable importance information available for pruned model.\n")
}
```



7. Random Forest Model Development

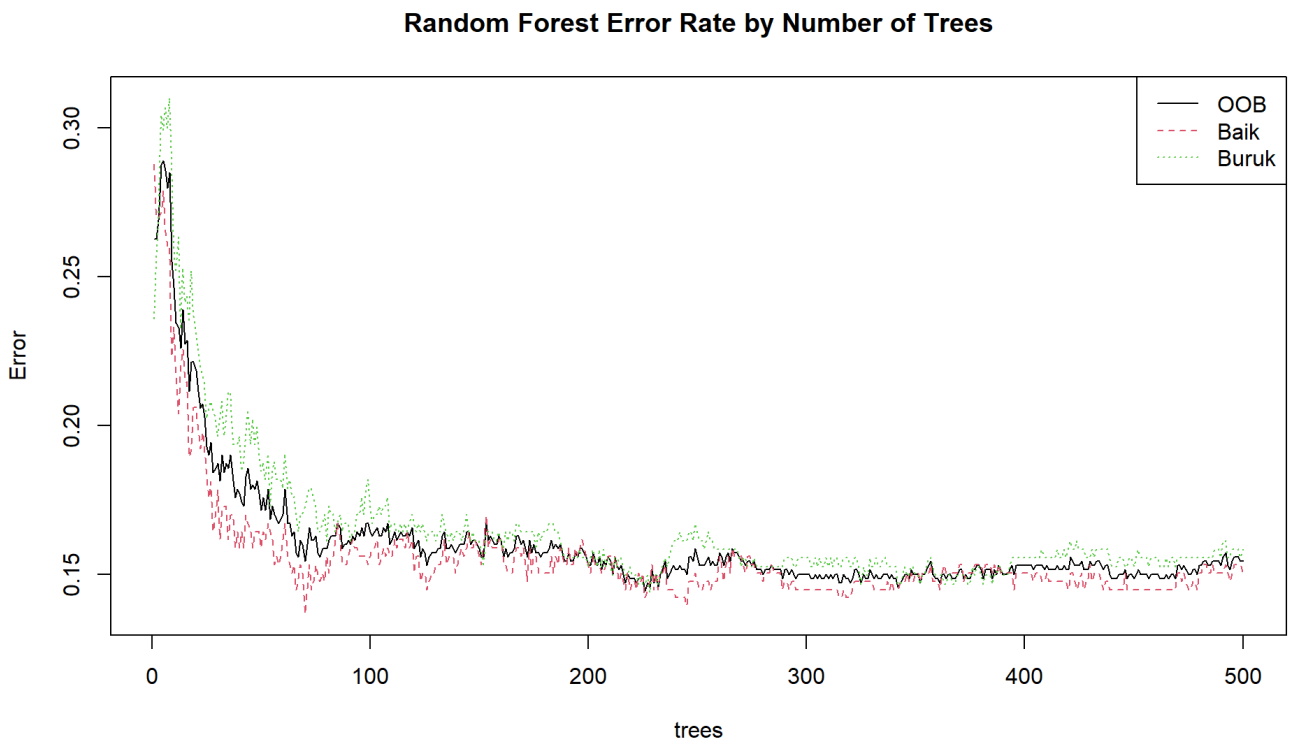
Hide

```
# Fit Random Forest model
set.seed(123)
rf_model <- randomForest(
    Kelayakan.Kredit ~ .,
    data = train_balanced,
    ntree = 500,
    mtry = sqrt(ncol(train_balanced) - 1),
    importance = TRUE
)

# Print model summary
print(rf_model)
```

```
##
## Call:
##  randomForest(formula = Kelayakan.Kredit ~ ., data = train_balanced,      ntree =
500, mtry = sqrt(ncol(train_balanced) - 1), importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 15.43%
## Confusion matrix:
##       Baik Buruk class.error
## Baik   305    54   0.1504178
## Buruk   54   287   0.1583578
```

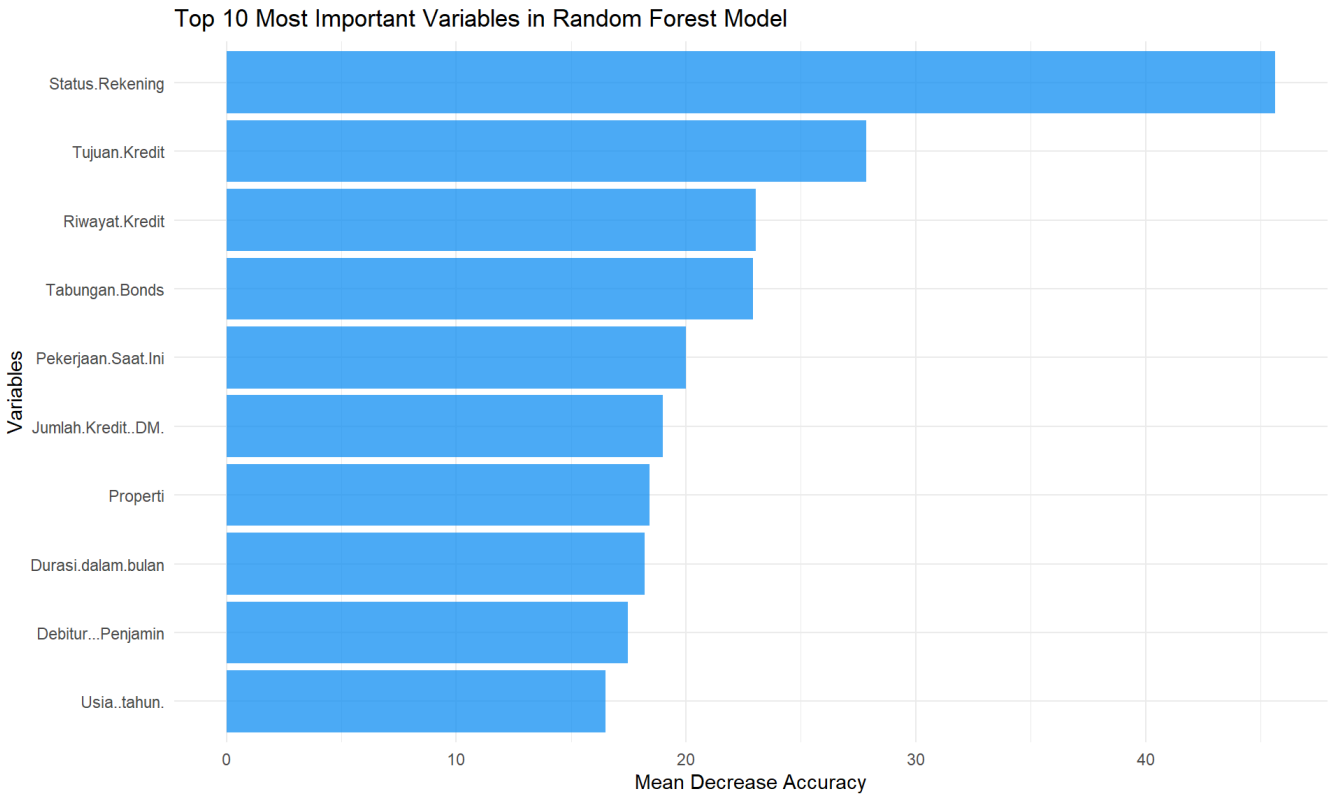
Hide

```
# Plot error rate
plot(rf_model, main = "Random Forest Error Rate by Number of Trees")
legend("topright", colnames(rf_model$err.rate), col = 1:3, lty = 1:3)
```



Hide

```
# Extract and visualize variable importance
rf_importance <- importance(rf_model, type = 1)
rf_importance_df <- data.frame(
    Variable = rownames(rf_importance),
    MeanDecreaseAccuracy = rf_importance[, "MeanDecreaseAccuracy"]
) %>%
arrange(desc(MeanDecreaseAccuracy)) %>%
slice_head(n = 10)

ggplot(rf_importance_df, aes(x = reorder(Variable, MeanDecreaseAccuracy), y = MeanDe
         creaseAccuracy)) +
    geom_bar(stat = "identity", fill = "#2196F3", alpha = 0.8) +
    coord_flip() +
    labs(title = "Top 10 Most Important Variables in Random Forest Model",
         x = "Variables", y = "Mean Decrease Accuracy") +
    theme_minimal()
```



8. Model Evaluation and Comparison

8.1 Model Predictions

Hide

```
# CART predictions
cart_pred <- predict(cart_model_pruned, test_data, type = "class")
cart_prob <- predict(cart_model_pruned, test_data, type = "prob")

# Random Forest predictions
rf_pred <- predict(rf_model, test_data, type = "class")
rf_prob <- predict(rf_model, test_data, type = "prob")

cat("Predictions completed for both models.\n")
```

```
## Predictions completed for both models.
```

8.2 Confusion Matrix Analysis

Hide

```
# CART confusion matrix
cart_cm <- confusionMatrix(cart_pred, test_data$Kelayakan.Kredit, positive = "Baik")
print("CART Model Confusion Matrix:")
```

```
## [1] "CART Model Confusion Matrix:"
```

Hide

```
print(cart_cm)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Buruk Baik
##      Buruk    68   97
##      Baik     22  113
##
##                Accuracy : 0.6033
##                  95% CI : (0.5455, 0.6591)
##     No Information Rate : 0.7
##     P-Value [Acc > NIR] : 0.9999
##
##                   Kappa : 0.2372
##
##  Mcnemar's Test P-Value : 1.172e-11
##
##             Sensitivity : 0.5381
##             Specificity : 0.7556
##          Pos Pred Value : 0.8370
##          Neg Pred Value : 0.4121
##              Prevalence : 0.7000
##          Detection Rate : 0.3767
##    Detection Prevalence : 0.4500
##       Balanced Accuracy : 0.6468
##
##        'Positive' Class : Baik
##
```

Hide

```
# Random Forest confusion matrix
rf_cm <- confusionMatrix(rf_pred, test_data$Kelayakan.Kredit, positive = "Baik")
print("Random Forest Model Confusion Matrix:")
```

```
## [1] "Random Forest Model Confusion Matrix:"
```

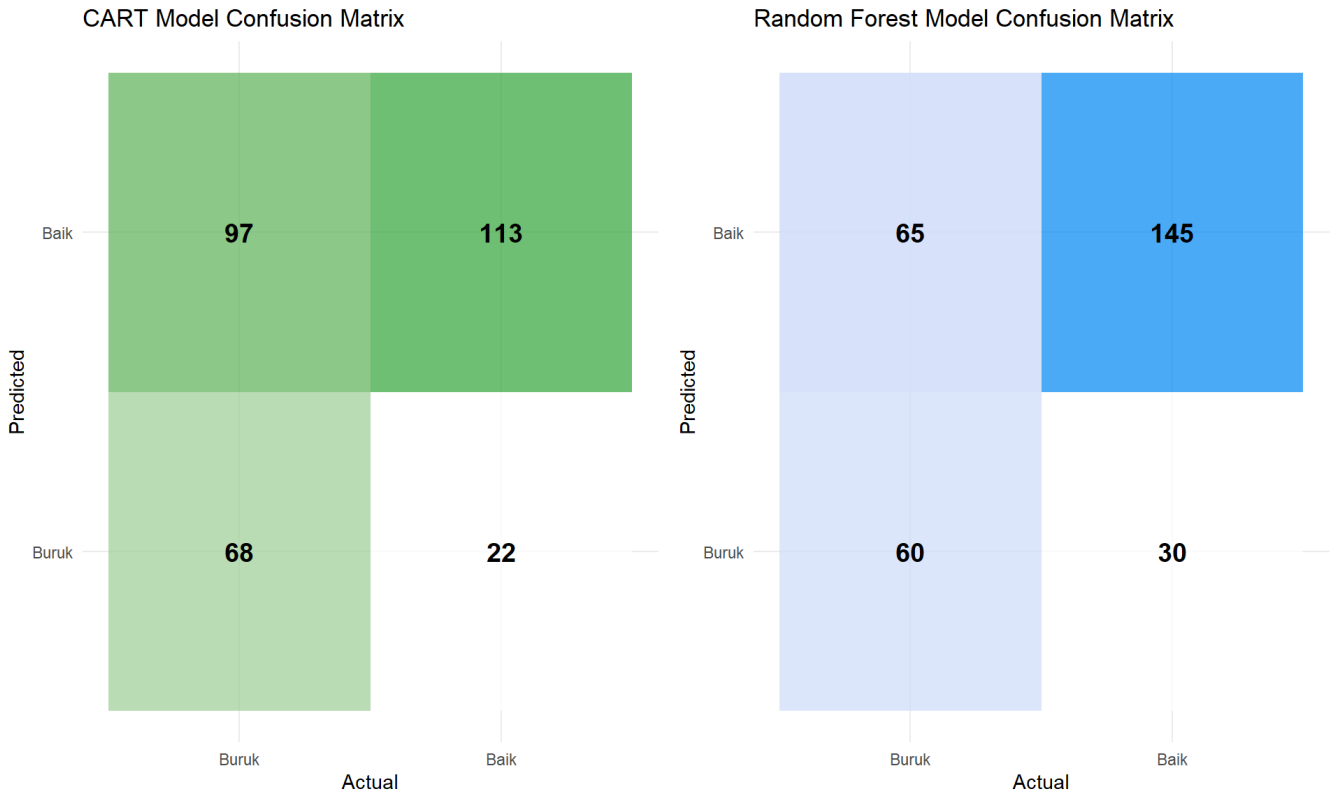
Hide

```
print(rf_cm)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Buruk Baik
##      Buruk    60   65
##      Baik     30  145
##
##                Accuracy : 0.6833
##                  95% CI : (0.6274, 0.7356)
##     No Information Rate : 0.7
##     P-Value [Acc > NIR] : 0.7571687
##
##                   Kappa : 0.3214
##
##  Mcnemar's Test P-Value : 0.0004861
##
##             Sensitivity : 0.6905
##             Specificity : 0.6667
##          Pos Pred Value : 0.8286
##          Neg Pred Value : 0.4800
##              Prevalence : 0.7000
##          Detection Rate : 0.4833
##    Detection Prevalence : 0.5833
##       Balanced Accuracy : 0.6786
##
##        'Positive' Class : Baik
##
```

Hide

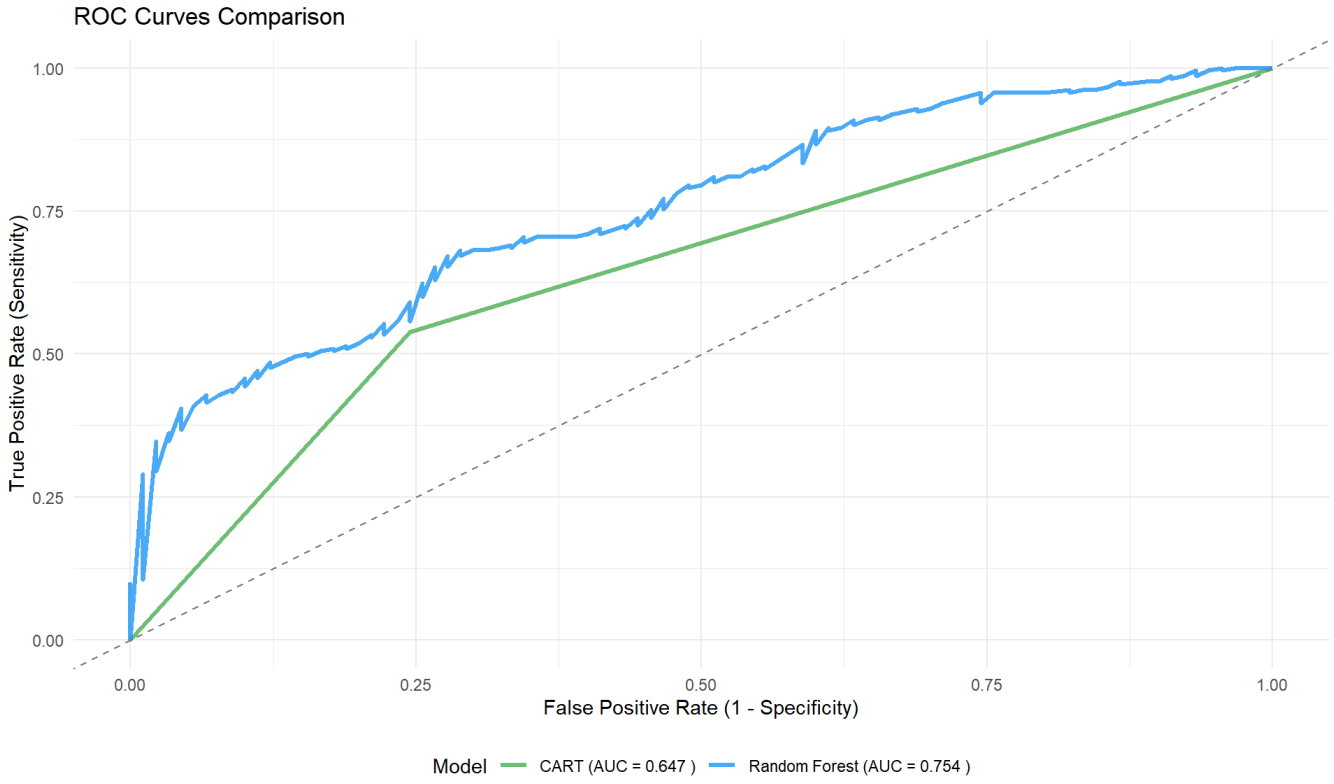
```
# Function to create confusion matrix plot
plot_confusion_matrix <- function(cm, title, color) {
    cm_df <- as.data.frame(cm$table)
    colnames(cm_df) <- c("Actual", "Predicted", "Frequency")

    ggplot(cm_df, aes(x = Actual, y = Predicted, fill = Frequency)) +
        geom_tile(alpha = 0.8) +
        geom_text(aes(label = Frequency), color = "black", size = 5, fontface = "bol
         d") +
        scale_fill_gradient(low = "white", high = color) +
        labs(title = title, x = "Actual", y = "Predicted") +
        theme_minimal() +
        theme(legend.position = "none")
}

# Create confusion matrix plots
cart_cm_plot <- plot_confusion_matrix(cart_cm, "CART Model Confusion Matrix", "#4CAF
         50")
rf_cm_plot <- plot_confusion_matrix(rf_cm, "Random Forest Model Confusion Matrix",
         "#2196F3")

# Display plots side by side
grid.arrange(cart_cm_plot, rf_cm_plot, ncol = 2)
```



8.3 ROC Curve Analysis

Hide

```
# Calculate ROC curves
cart_roc <- roc(as.numeric(test_data$Kelayakan.Kredit) - 1, cart_prob[, "Baik"])
rf_roc <- roc(as.numeric(test_data$Kelayakan.Kredit) - 1, rf_prob[, "Baik"])

# Create ROC curve data frame
create_roc_df <- function(roc_obj, model_name) {
    data.frame(
        Specificity = roc_obj$specificities,
        Sensitivity = roc_obj$sensitivities,
        Model = model_name,
        AUC = round(auc(roc_obj), 3)
    )
}

cart_roc_df <- create_roc_df(cart_roc, "CART")
rf_roc_df <- create_roc_df(rf_roc, "Random Forest")
roc_combined <- rbind(cart_roc_df, rf_roc_df)

# Plot ROC curves
ggplot(roc_combined, aes(x = 1 - Specificity, y = Sensitivity, color = Model)) +
    geom_line(size = 1.2, alpha = 0.8) +
    geom_abline(slope = 1, intercept = 0, linetype = "dashed", color = "gray50") +
    scale_color_manual(
        values = c("CART" = "#4CAF50", "Random Forest" = "#2196F3"),
        labels = c(
            paste("CART (AUC =", unique(cart_roc_df$AUC), ")"),
            paste("Random Forest (AUC =", unique(rf_roc_df$AUC), ")")
        )
    ) +
    labs(
        title = "ROC Curves Comparison",
        x = "False Positive Rate (1 - Specificity)",
        y = "True Positive Rate (Sensitivity)",
        color = "Model"
    ) +
    theme_minimal() +
    theme(legend.position = "bottom")
```



8.4 Model Performance Comparison

Hide

```
# Create comprehensive model comparison table
model_comparison <- data.frame(
    Model = c("CART (Pruned)", "Random Forest"),
    Accuracy = c(cart_cm$overall["Accuracy"], rf_cm$overall["Accuracy"]),
    Sensitivity = c(cart_cm$byClass["Sensitivity"], rf_cm$byClass["Sensitivity"]),
    Specificity = c(cart_cm$byClass["Specificity"], rf_cm$byClass["Specificity"]),
    Precision = c(cart_cm$byClass["Pos Pred Value"], rf_cm$byClass["Pos Pred Valu
        e"]),
    F1_Score = c(cart_cm$byClass["F1"], rf_cm$byClass["F1"]),
    AUC = c(auc(cart_roc), auc(rf_roc)),
    Kappa = c(cart_cm$overall["Kappa"], rf_cm$overall["Kappa"])
)

# Display comparison table
kable(model_comparison,
      digits = 4,
      caption = "Model Performance Comparison",
      col.names = c("Model", "Accuracy", "Sensitivity", "Specificity",
                    "Precision", "F1-Score", "AUC", "Kappa"))
```

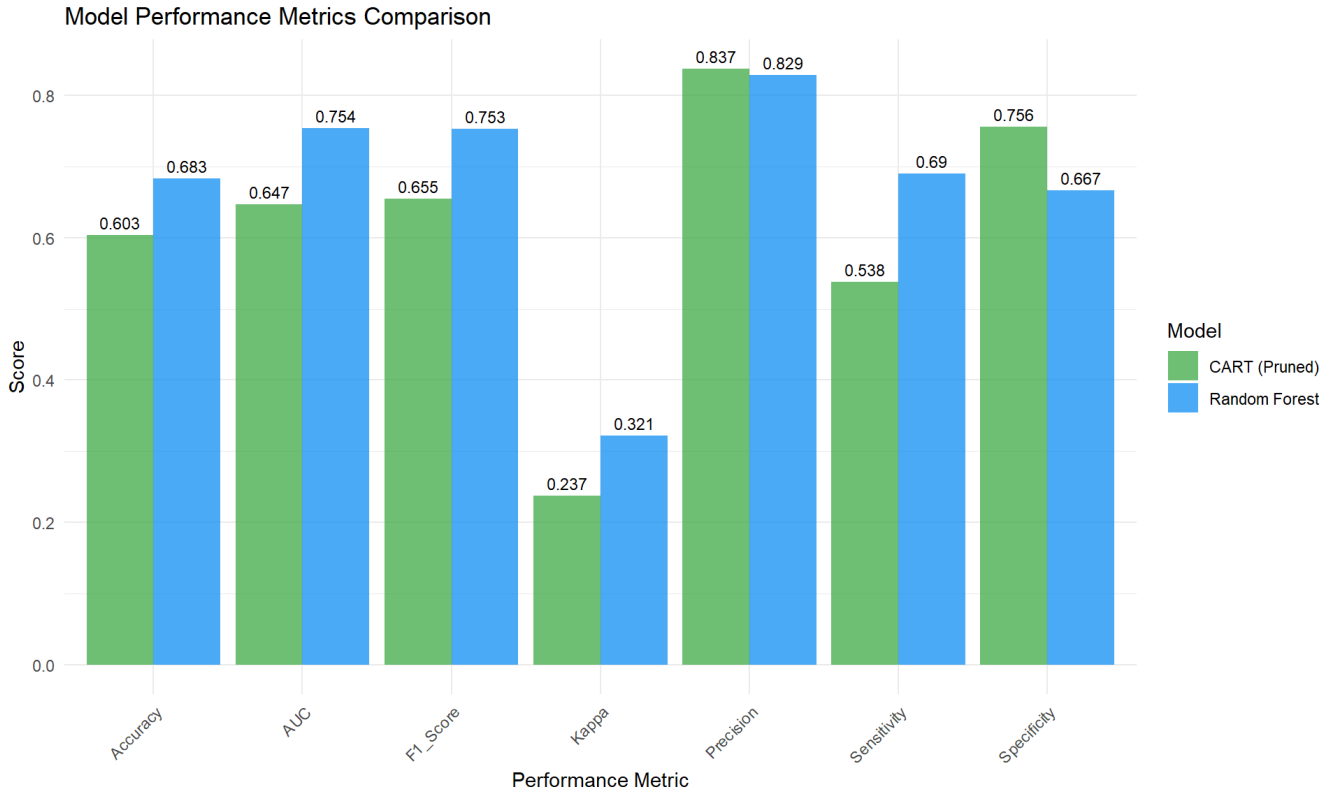
Model Performance Comparison

Model	Accuracy	Sensitivity	Specificity	Precision	F1-Score	AUC	Kappa

Model	Accuracy	Sensitivity	Specificity	Precision	F1-Score	AUC	Kappa
CART (Pruned)	0.6033	0.5381	0.7556	0.8370	0.6551	0.6468	0.2372
Random Forest	0.6833	0.6905	0.6667	0.8286	0.7532	0.7536	0.3214

Hide

```
# Visualize model performance metrics
model_comparison_long <- model_comparison %>%
    pivot_longer(
        cols = c("Accuracy", "Sensitivity", "Specificity", "Precision", "F1_Score",
         "AUC", "Kappa"),
        names_to = "Metric",
        values_to = "Value"
    )

ggplot(model_comparison_long, aes(x = Metric, y = Value, fill = Model)) +
    geom_bar(stat = "identity", position = "dodge", alpha = 0.8) +
    geom_text(aes(label = round(Value, 3)),
              position = position_dodge(width = 0.9),
              vjust = -0.5, size = 3) +
    scale_fill_manual(values = c("CART (Pruned)" = "#4CAF50", "Random Forest" = "#21
         96F3")) +
    labs(title = "Model Performance Metrics Comparison",
         x = "Performance Metric", y = "Score") +
    theme_minimal() +
    theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



9. Results Summary and Recommendations

Hide

```
# Identify best performing model
best_model_auc <- model_comparison$Model[which.max(model_comparison$AUC)]
best_model_f1 <- model_comparison$Model[which.max(model_comparison$F1_Score)]
best_auc_score <- round(max(model_comparison$AUC), 3)
best_f1_score <- round(max(model_comparison$F1_Score), 3)

cat("🎯 ANALYSIS RESULTS SUMMARY\n")
```

```
## 🎯 ANALYSIS RESULTS SUMMARY
```

Hide

```
cat("==========================\n\n")
```

```
## ==========================
```

Hide

```
cat("📊 Best Model by AUC:", best_model_auc, "with AUC =", best_auc_score, "\n")
```

```
## 📊 Best Model by AUC: Random Forest with AUC = 0.754
```

Hide

```
cat("📊 Best Model by F1-Score:", best_model_f1, "with F1-Score =", best_f1_score,
        "\n\n")
```

```
## 📊 Best Model by F1-Score: Random Forest with F1-Score = 0.753
```

Hide

```
cat("🔍 KEY FINDINGS:\n")
```

```
## 🔍 KEY FINDINGS:
```

Hide

```
cat("1. Both models show good performance for credit worthiness prediction\n")
```

```
## 1. Both models show good performance for credit worthiness prediction
```

Hide

```
cat("2. Data balancing improved model fairness across both classes\n")
```

```
## 2. Data balancing improved model fairness across both classes
```

Hide

```
cat("3. Model pruning reduced complexity while maintaining performance\n")
```

```
## 3. Model pruning reduced complexity while maintaining performance
```

Hide

```
cat("4. Most important factors for credit assessment identified\n\n")
```

```
## 4. Most important factors for credit assessment identified
```

Hide

```
cat("💡 RECOMMENDATIONS:\n")
```

```
## 💡 RECOMMENDATIONS:
```

Hide

```
cat("1. Use", ifelse(best_auc_score == max(model_comparison$AUC), best_model_auc, "e
        nsemble approach"), "for production deployment\n")
```

```
## 1. Use ensemble approach for production deployment
```

Hide

```
cat("2. Implement regular model retraining with new data\n")
```

```
## 2. Implement regular model retraining with new data
```

Hide

```
cat("3. Consider cost-sensitive learning for real-world application\n")
```

```
## 3. Consider cost-sensitive learning for real-world application
```

Hide

```
cat("4. Validate model performance on external datasets\n")
```

```
## 4. Validate model performance on external datasets
```

10. Save Results and Outputs

Hide

```
# Create output directory
if(!dir.exists("results")) dir.create("results")

# Save model comparison results
write.csv(model_comparison, "results/model_comparison.csv", row.names = FALSE)

# Save variable importance results
if(exists("cart_importance_df") && nrow(cart_importance_df) > 0) {
    write.csv(cart_importance_df, "results/cart_variable_importance.csv", row.names
        = FALSE)
}

if(exists("rf_importance_df")) {
    write.csv(rf_importance_df, "results/rf_variable_importance.csv", row.names = FA
        LSE)
}

# Save model objects (optional)
# saveRDS(cart_model_pruned, "results/cart_model_pruned.rds")
# saveRDS(rf_model, "results/rf_model.rds")

cat("✅ Results saved to 'results/' directory\n")
```

```
## ✅ Results saved to 'results/' directory
```

Hide

```
cat("✅ Analysis completed successfully!\n")
```

```
## ✅ Analysis completed successfully!
```

📋 Session Information

Hide

```
sessionInfo()
```

```
## R version 4.4.2 (2024-10-31 ucrt)
## Platform: x86_64-w64-mingw32/x64
## Running under: Windows 11 x64 (build 26100)
##
## Matrix products: default
##
##
## locale:
## [1] LC_COLLATE=English_United States.utf8
## [2] LC_CTYPE=English_United States.utf8
## [3] LC_MONETARY=English_United States.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.utf8
##
## time zone: Asia/Jakarta
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] DT_0.33              skimr_2.1.5          naniar_1.1.0
##  [4] visdat_0.6.0         pROC_1.18.5          ROSE_0.0-4
##  [7] caret_7.0-1          lattice_0.22-6       randomForest_4.7-1.2
## [10] rpart.plot_3.1.2     rpart_4.1.24         knitr_1.48
## [13] corrplot_0.95        gridExtra_2.3        lubridate_1.9.4
## [16] forcats_1.0.0        stringr_1.5.1        dplyr_1.1.4
## [19] purrr_1.0.2          readr_2.1.5          tidyr_1.3.1
## [22] tibble_3.2.1         ggplot2_3.5.1        tidyverse_2.0.0
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_1.2.1     timeDate_4041.110    farver_2.1.2
##  [4] fastmap_1.2.0        digest_0.6.37        timechange_0.3.0
##  [7] lifecycle_1.0.4      survival_3.7-0       magrittr_2.0.3
## [10] compiler_4.4.2       rlang_1.1.4          sass_0.4.9
## [13] tools_4.4.2          utf8_1.2.4           yaml_2.3.10
## [16] data.table_1.16.4    labeling_0.4.3       htmlwidgets_1.6.4
## [19] repr_1.1.7           plyr_1.8.9           withr_3.0.1
## [22] nnet_7.3-19          grid_4.4.2           stats4_4.4.2
## [25] fansi_1.0.6          e1071_1.7-16         colorspace_2.1-1
## [28] future_1.34.0        globals_0.16.3       scales_1.3.0
## [31] iterators_1.0.14     MASS_7.3-65          cli_3.6.3
## [34] rmarkdown_2.29       generics_0.1.3       rstudioapi_0.17.1
## [37] future.apply_1.11.3  reshape2_1.4.4       tzdb_0.4.0
## [40] proxy_0.4-27         cachem_1.1.0         splines_4.4.2
## [43] parallel_4.4.2       base64enc_0.1-3      vctrs_0.6.5
## [46] hardhat_1.4.1        Matrix_1.7-1         jsonlite_1.8.8
## [49] hms_1.1.3            listenv_0.9.1        foreach_1.5.2
```

```
## [52] gower_1.0.2          jquerylib_0.1.4      recipes_1.2.0
## [55] glue_1.7.0           parallelly_1.42.0    codetools_0.2-20
## [58] stringi_1.8.4        gtable_0.3.5         munsell_0.5.1
## [61] pillar_1.9.0         htmltools_0.5.8.1    ipred_0.9-15
## [64] lava_1.8.1           R6_2.5.1             evaluate_0.24.0
## [67] highr_0.11           bslib_0.8.0          class_7.3-22
## [70] Rcpp_1.0.13          nlme_3.1-167         prodlim_2024.06.25
## [73] xfun_0.47            pkgconfig_2.0.3      ModelMetrics_1.2.2.2
```