Credit Risk EDA and Modeling

Code

Jestin

2025-08-13

Step 1: Load Data

Hide

```
file_path <- "/Users/Jestin/Desktop/Credit Risk Project/credit_risk_dataset.csv"

if (!file.exists(file_path)) {
  stop("CSV file not found at the specified path: ", file_path)
}

data <- tryCatch(
  read.csv(file_path, stringsAsFactors = FALSE),
  error = function(e) stop("Error reading file: ", e$message)
)

cat("Data loaded:", nrow(data), "rows and", ncol(data), "columns\n")
```

```
Data loaded: 32574 rows and 12 columns
```

Step 2: Overview of Dataset

Hide

```
skim(data)
```

```
── Data Summary ────────────────────────
                           Values
Name                       data
Number of rows             32574
Number of columns          12
_________________________
Column type frequency:
  character                4
  numeric                  8
_________________________
Group variables            None
```

skim_variable	n_missing	complete_rate	m.	m.	em…	n_unique	whitespace
<chr>	<int>	<dbl>	<int>	<int>	<int>	<int>	<int>
1 person_home_ownership	0	1	3	8	0	4	0

	skim_variable	n_missing	complete_rate	m.	m.	em...	n_unique	whitespace
	<chr>	<int>	<dbl>	<int>	<int>	<int>	<int>	<int>
2	loan_intent	0	1	7	17	0	6	0
3	loan_grade	0	1	1	1	0	7	0
4	cb_person_default_on_file	0	1	1	1	0	2	0

4 rows

	skim_variable	n_missing	complete_rate	mean	sd	
	<chr>	<int>	<dbl>	<dbl>	<dbl>	<db
1	person_age	0	1.0000000	2.771843e+01	6.204987e+00	20.(
2	person_income	0	1.0000000	6.587848e+04	5.253194e+04	4000.(
3	person_emp_length	895	0.9725241	4.782064e+00	4.034948e+00	0.(
4	loan_amnt	0	1.0000000	9.588018e+03	6.320250e+03	500.(
5	loan_int_rate	3115	0.9043716	1.101153e+01	3.240497e+00	5.4
6	loan_status	0	1.0000000	2.181801e-01	4.130167e-01	0.(
7	loan_percent_income	0	1.0000000	1.702017e-01	1.067549e-01	0.(
8	cb_person_cred_hist_length	0	1.0000000	5.804108e+00	4.053873e+00	2.(

8 rows | 1-8 of 11 columns

Hide

```
summary(data)
```

```
   person_age    person_income     person_home_ownership person_emp_length
 Min.   :20.00   Min.   :   4000   Length:32574          Min.   : 0.000
 1st Qu.:23.00   1st Qu.:  38500   Class :character      1st Qu.: 2.000
 Median :26.00   Median :  55000   Mode  :character      Median : 4.000
 Mean   :27.72   Mean   :  65878                         Mean   : 4.782
 3rd Qu.:30.00   3rd Qu.:  79200                         3rd Qu.: 7.000
 Max.   :94.00   Max.   :2039784                         Max.   :41.000
                                                         NA's   :895
 loan_intent         loan_grade          loan_amnt     loan_int_rate    loan_status
 Length:32574       Length:32574       Min.   :  500   Min.   : 5.42   Min.   :0.0000
 Class :character   Class :character   1st Qu.: 5000   1st Qu.: 7.90   1st Qu.:0.0000
 Mode  :character   Mode  :character   Median : 8000   Median :10.99   Median :0.0000
                                       Mean   : 9588   Mean   :11.01   Mean   :0.2182
                                       3rd Qu.:12200   3rd Qu.:13.47   3rd Qu.:0.0000
                                       Max.   :35000   Max.   :23.22   Max.   :1.0000
                                                       NA's   :3115
 loan_percent_income cb_person_default_on_file cb_person_cred_hist_length
 Min.   :0.0000      Length:32574              Min.   : 2.000
 1st Qu.:0.0900      Class :character          1st Qu.: 3.000
 Median :0.1500      Mode  :character          Median : 4.000
 Mean   :0.1702                                Mean   : 5.804
 3rd Qu.:0.2300                                3rd Qu.: 8.000
 Max.   :0.8300                                Max.   :30.000
```

Step 3: Data Cleaning

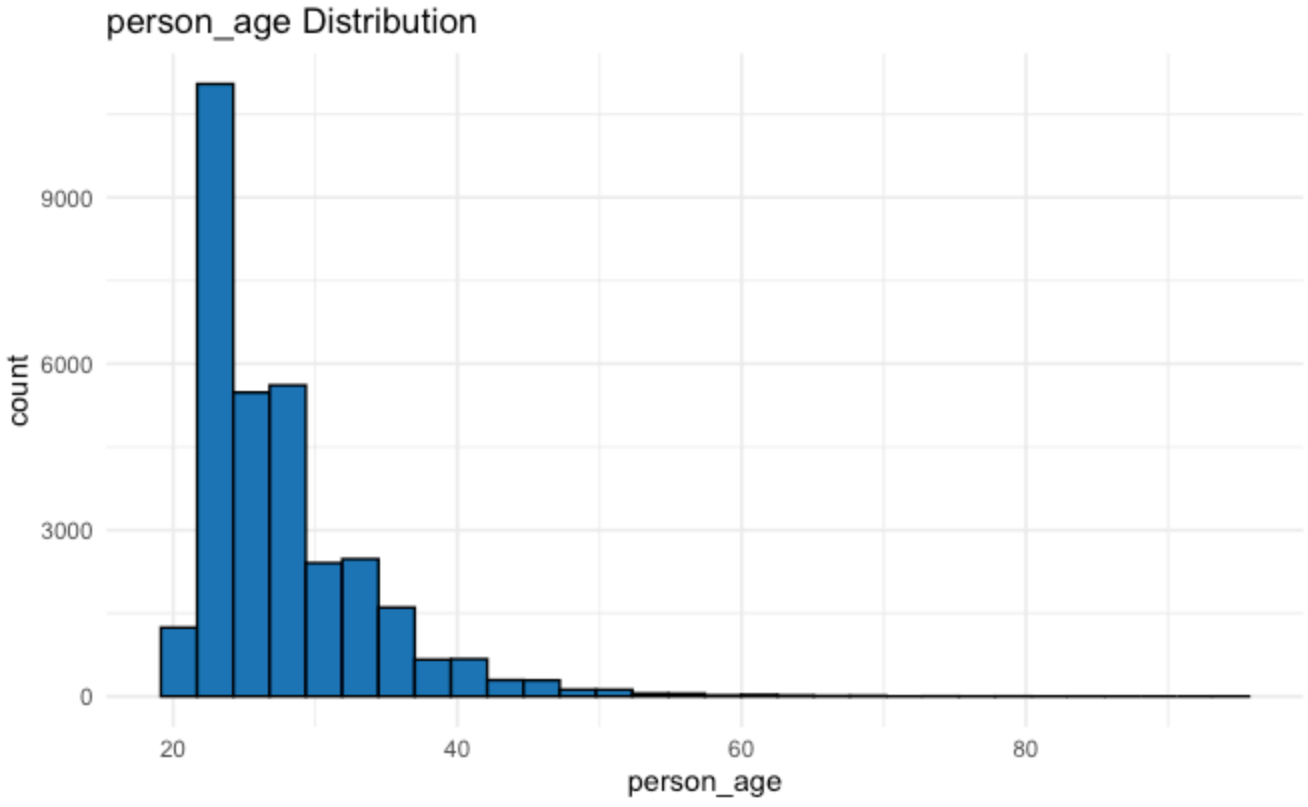
Hide

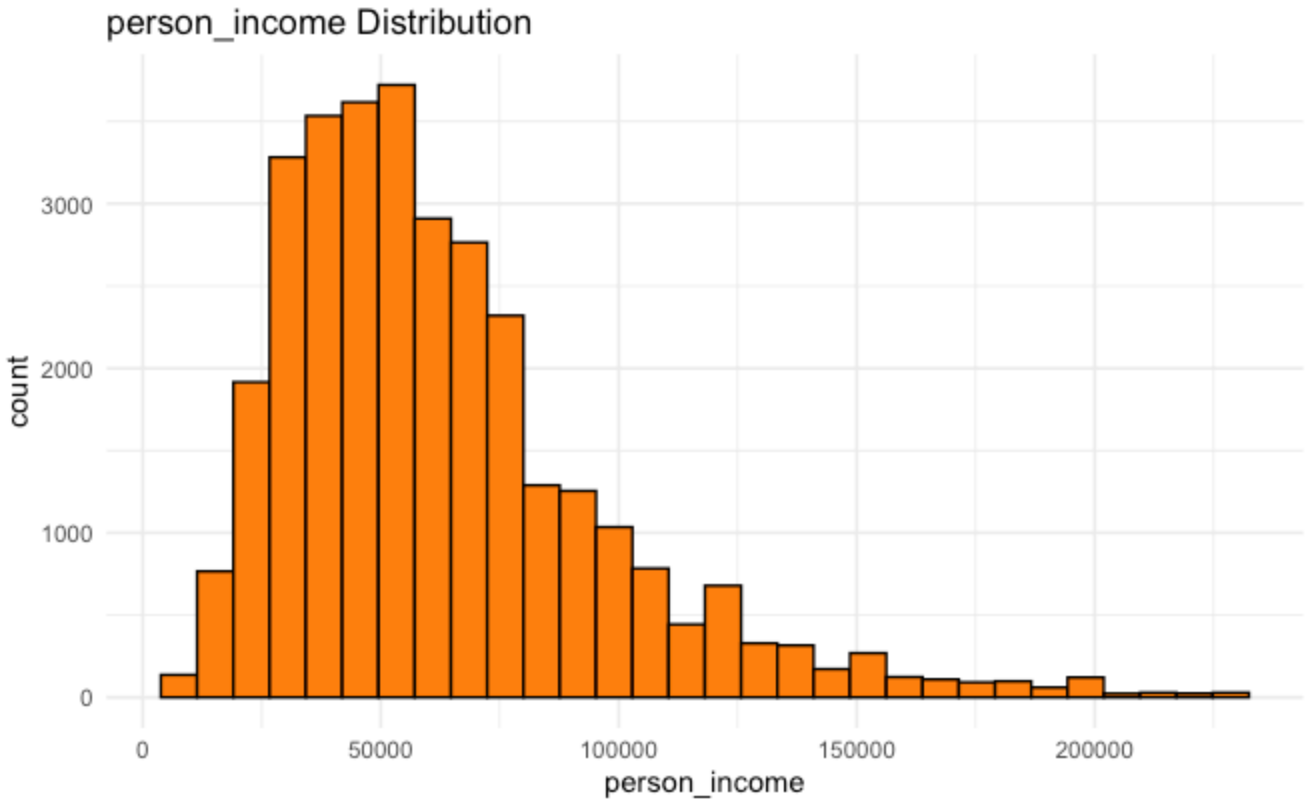
```
# Convert empty strings to NA
data <- data %>%
  mutate(across(where(is.character), ~ na_if(., "")))

# Impute numeric columns with median
num_cols <- c("loan_int_rate", "person_emp_length")
for (col in num_cols) {
  if (anyNA(data[[col]])) {
    data[[col]][is.na(data[[col]])] <- median(as.numeric(data[[col]]), na.rm = TRUE)
  }
}

# Filter unrealistic values
data <- data %>%
  filter(between(person_age, 18, 100)) %>%
  filter(person_income <= quantile(person_income, 0.99, na.rm = TRUE))
```

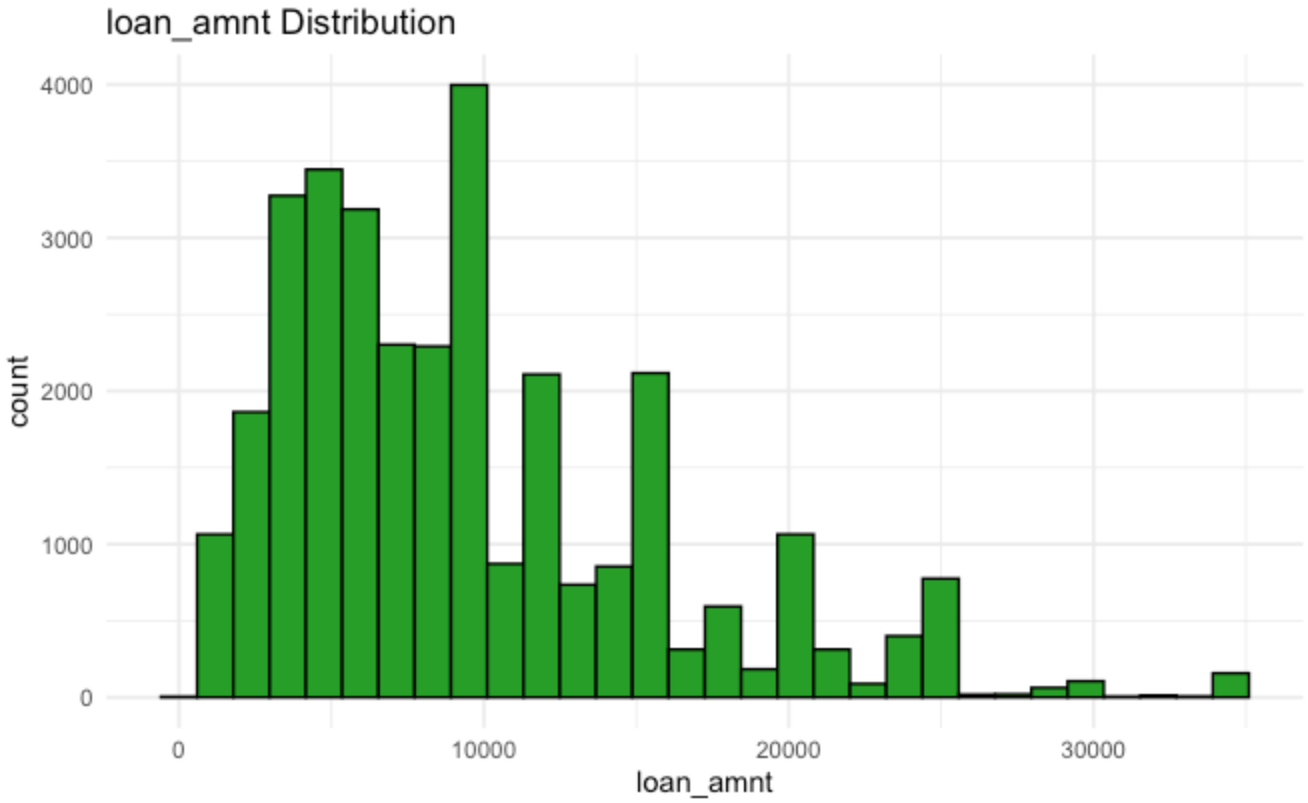
Step 4: Exploratory Data Analysis (EDA)

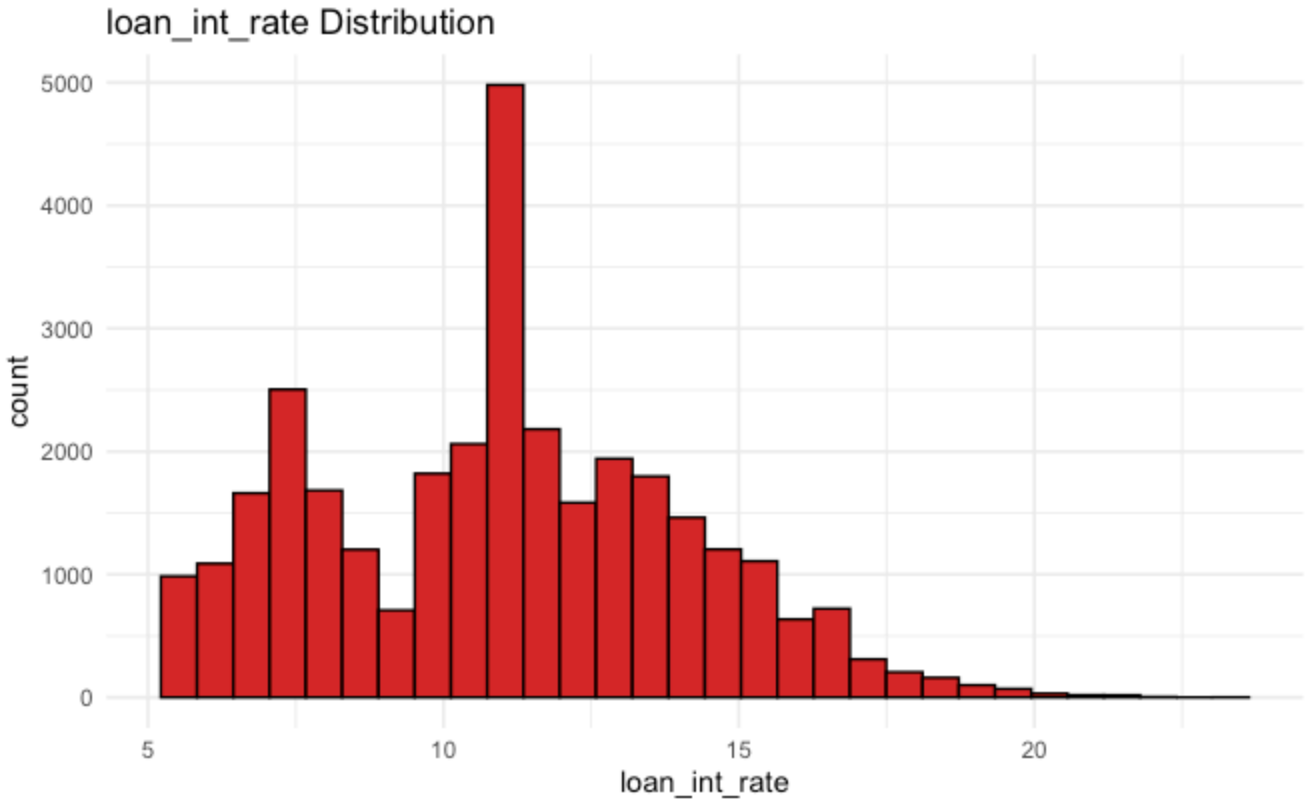
Hide

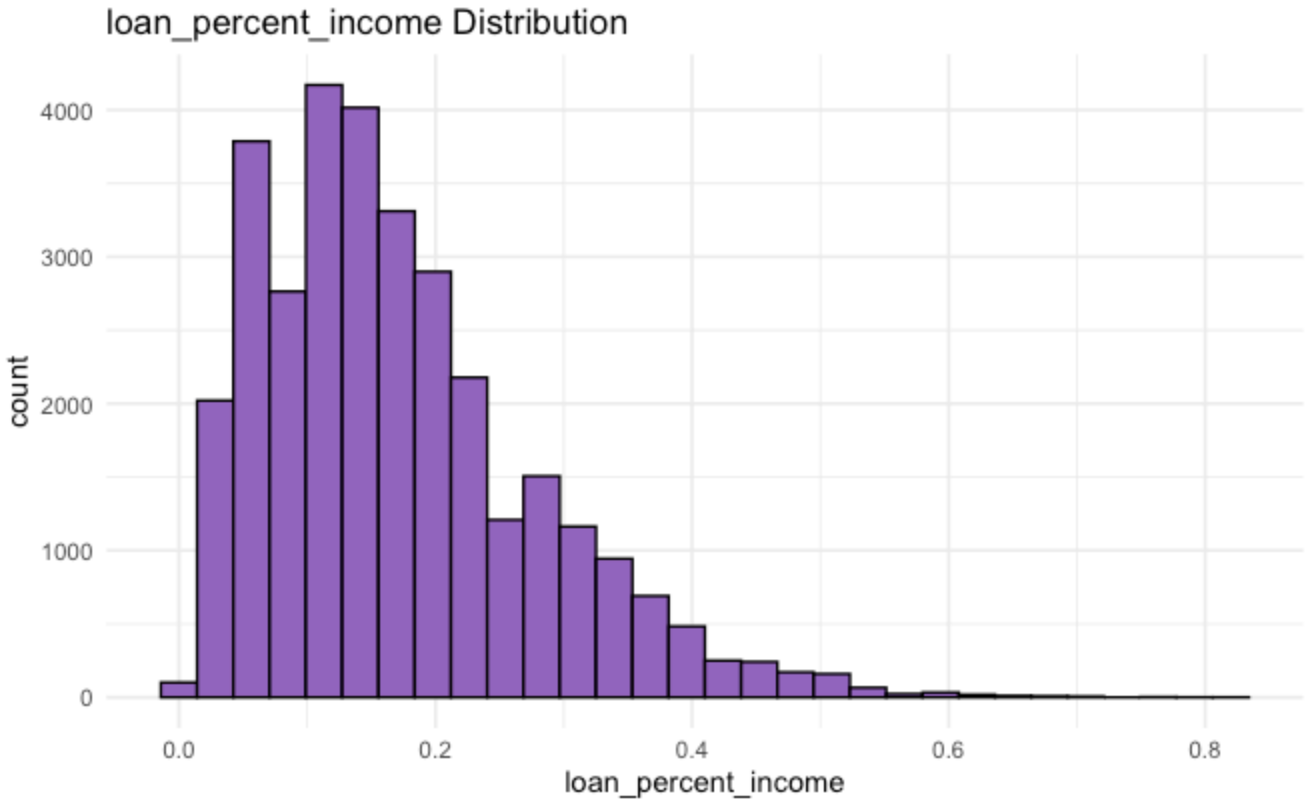
```
plot_hist <- function(df, var, fill_color) {
  ggplot(df, aes(x = .data[[var]])) +
    geom_histogram(bins = 30, fill = fill_color, color = "black") +
    ggtitle(paste(var, "Distribution")) +
    theme_minimal()
}

plot_bar <- function(df, var, fill_color) {
  ggplot(df, aes(x = .data[[var]])) +
    geom_bar(fill = fill_color) +
    ggtitle(paste(var, "Distribution")) +
    theme_minimal()
}

# Histograms
hist_vars <- c("person_age", "person_income", "loan_amnt", "loan_int_rate", "loan_percen
t_income")
hist_colors <- c("#1f77b4", "#ff7f0e", "#2ca02c", "#d62728", "#9467bd")
walk2(hist_vars, hist_colors, ~ print(plot_hist(data, .x, .y)))
```




person_income Distribution
count
3000
2000
1000
0
0
50000
100000
150000
200000
person_income

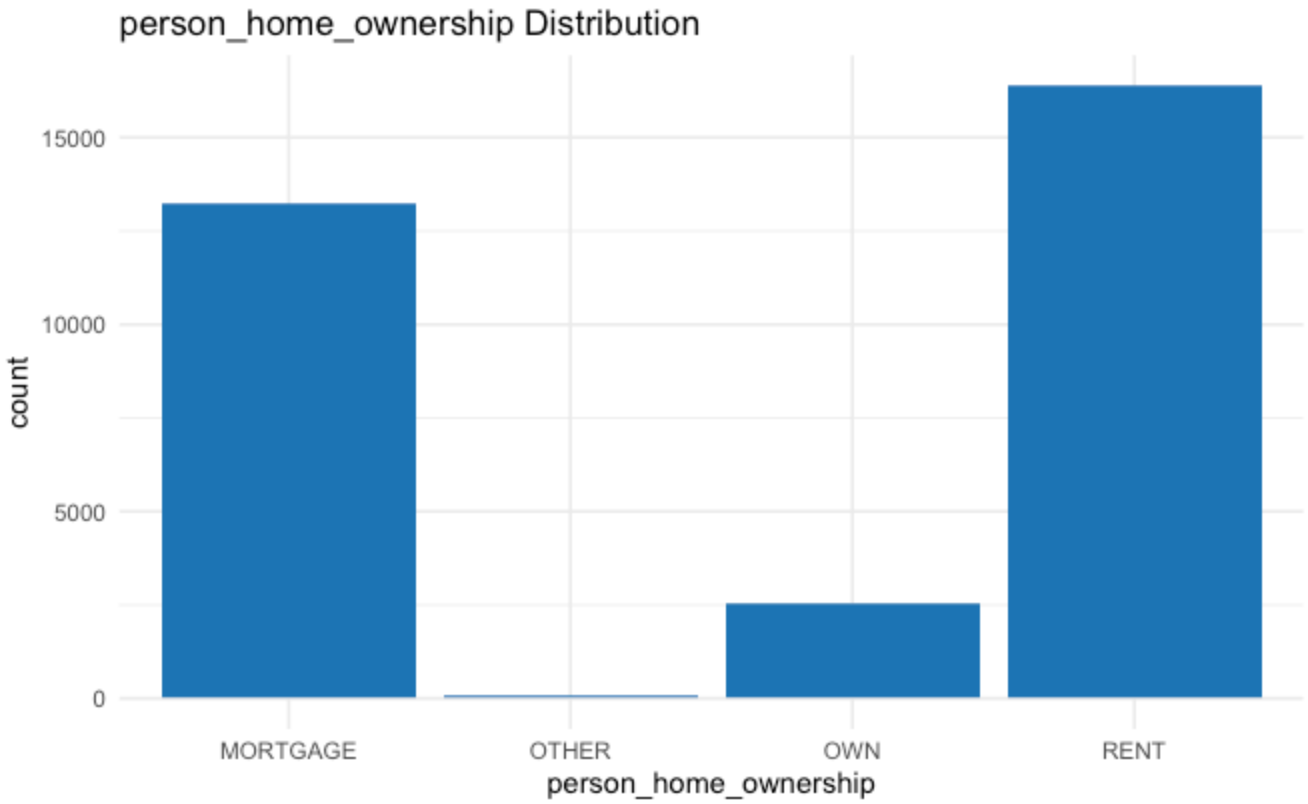

loan_amnt Distribution
count
4000
3000
2000
1000
0
0
10000
20000
30000
loan_amnt

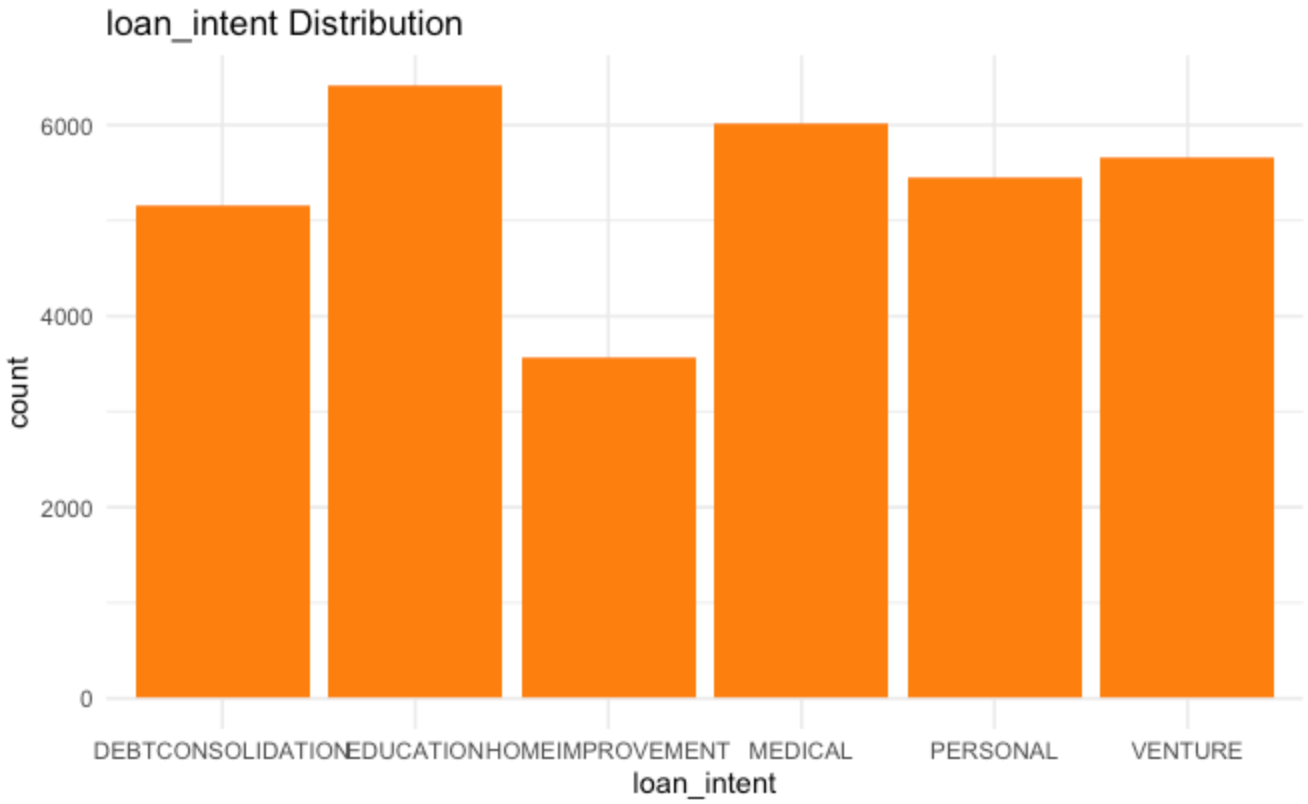

loan_int_rate Distribution
count
loan_int_rate
0
1000
2000
3000
4000
5000
5
10
15
20

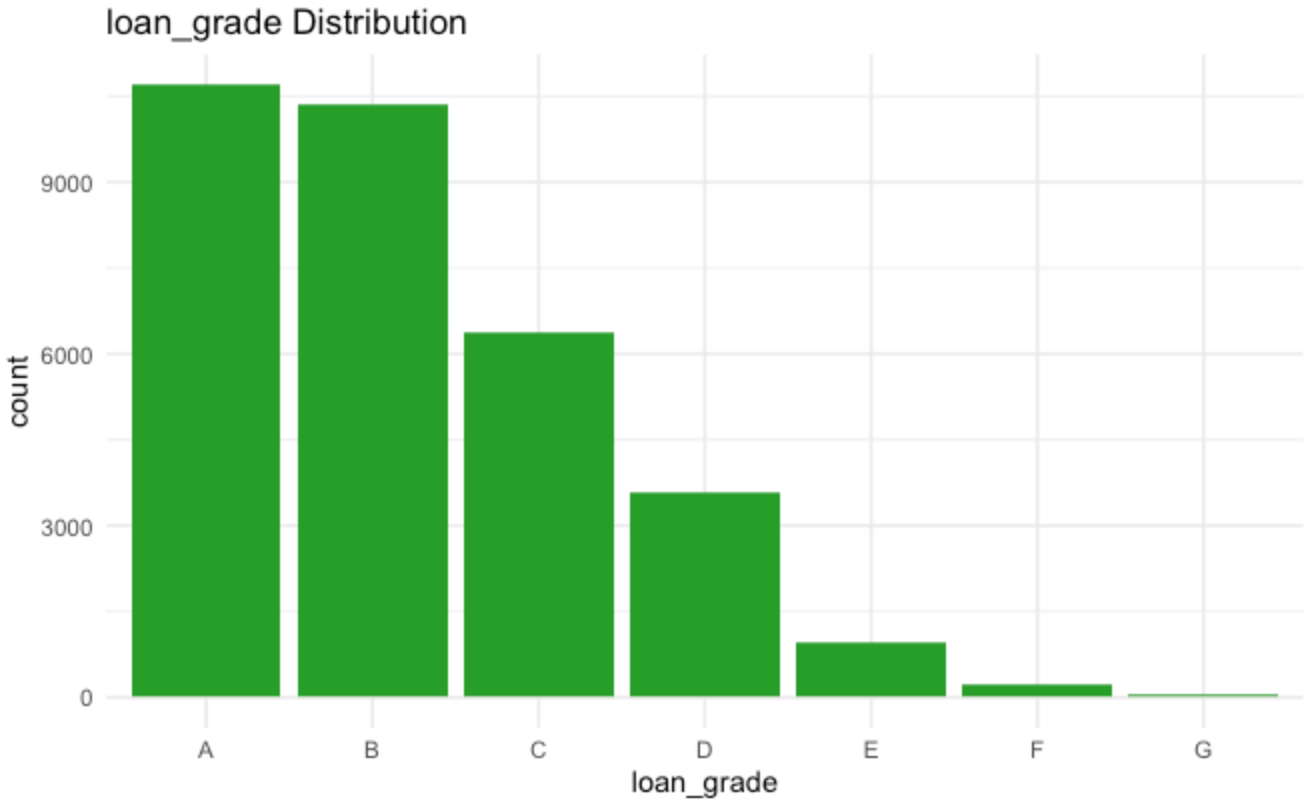

loan_percent_income Distribution
count
loan_percent_income
0
1000
2000
3000
4000
0.0
0.2
0.4
0.6
0.8

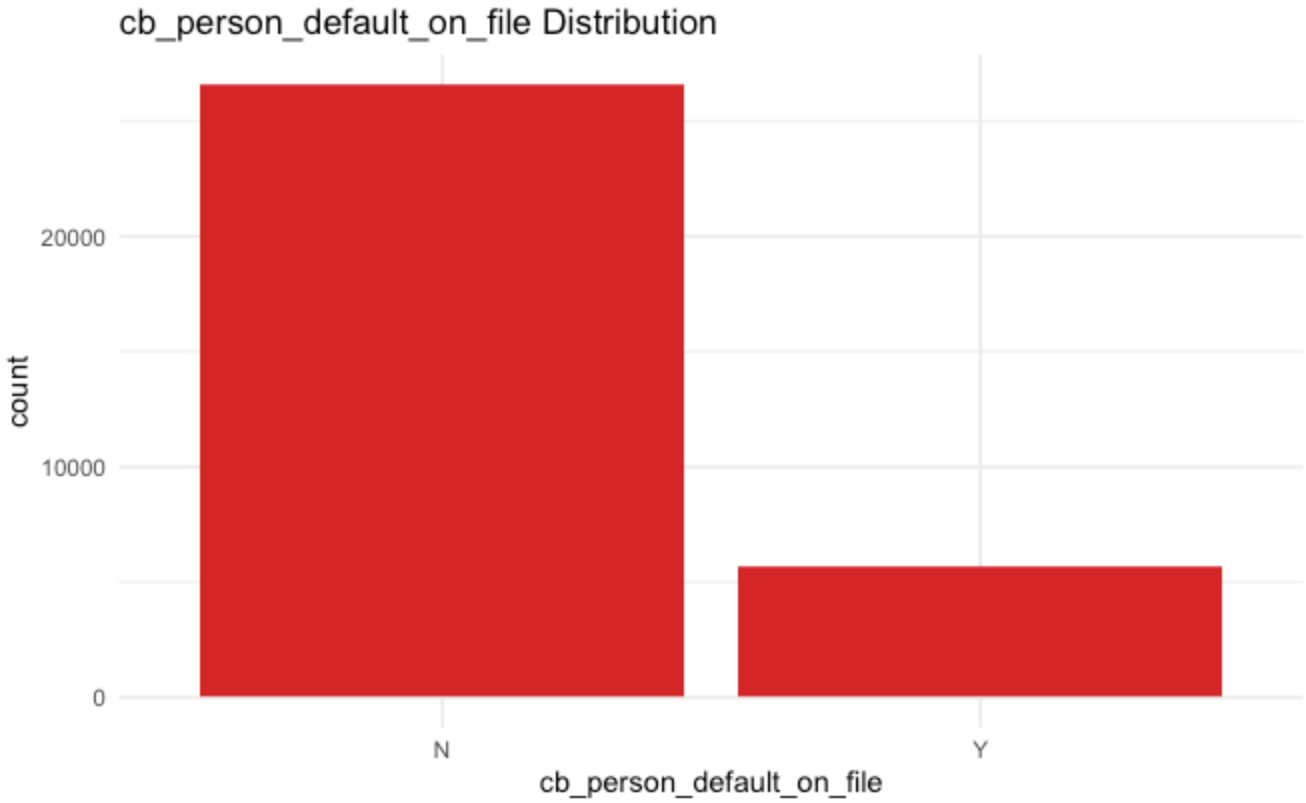
Hide

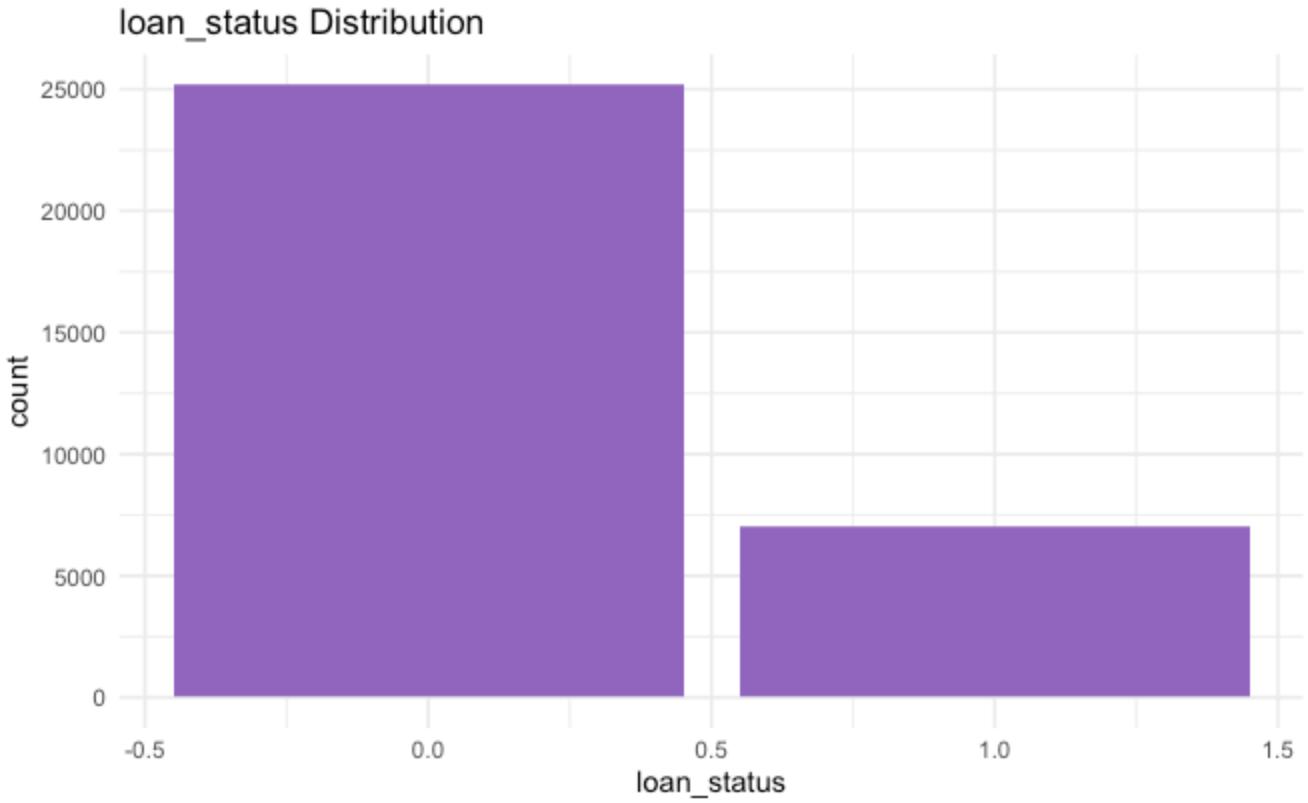
```
# Bar charts
bar_vars <- c("person_home_ownership", "loan_intent", "loan_grade", "cb_person_default_o
n_file", "loan_status")
bar_colors <- c("#1f77b4", "#ff7f0e", "#2ca02c", "#d62728", "#9467bd")
walk2(bar_vars, bar_colors, ~ print(plot_bar(data, .x, .y)))
```






loan_grade Distribution
count
9000
6000
3000
0
A
B
C
D
E
F
G
loan_grade

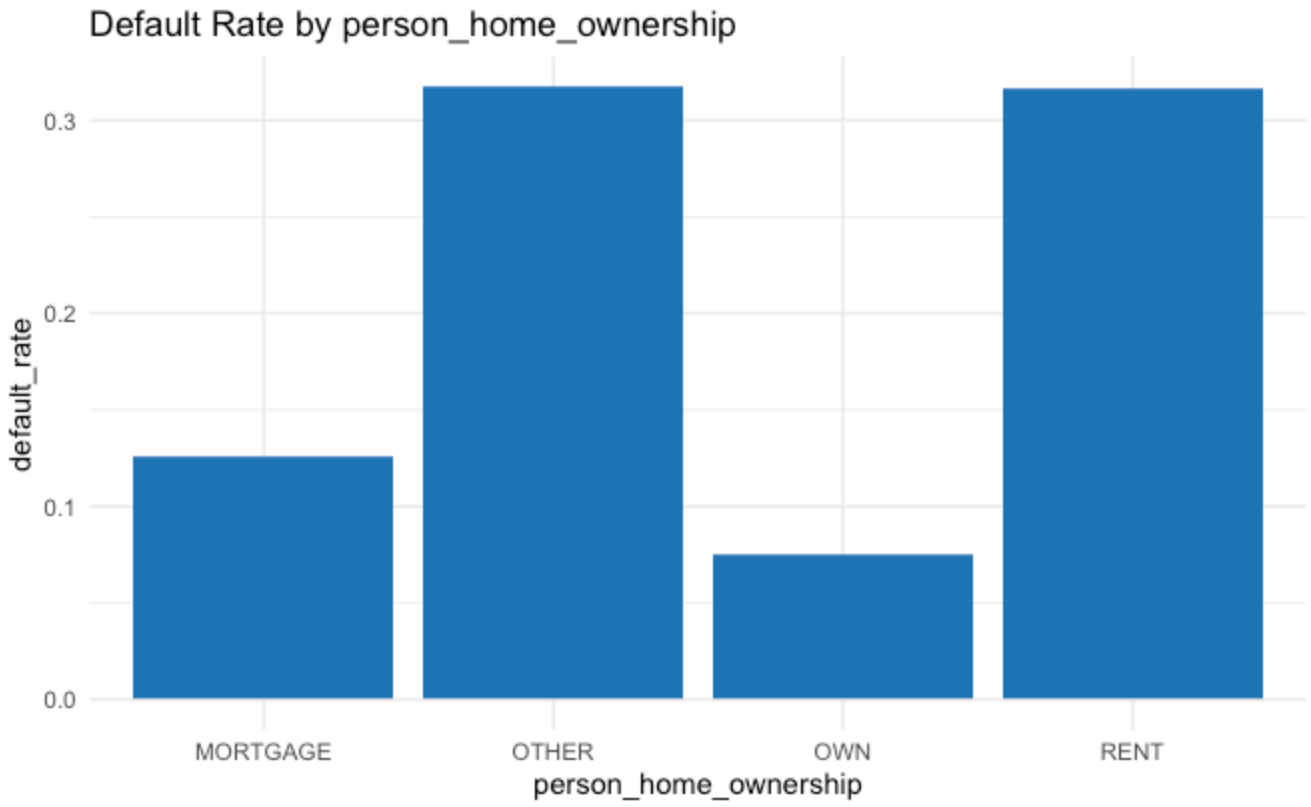

cb_person_default_on_file Distribution
count
20000
10000
0
N
Y
cb_person_default_on_file

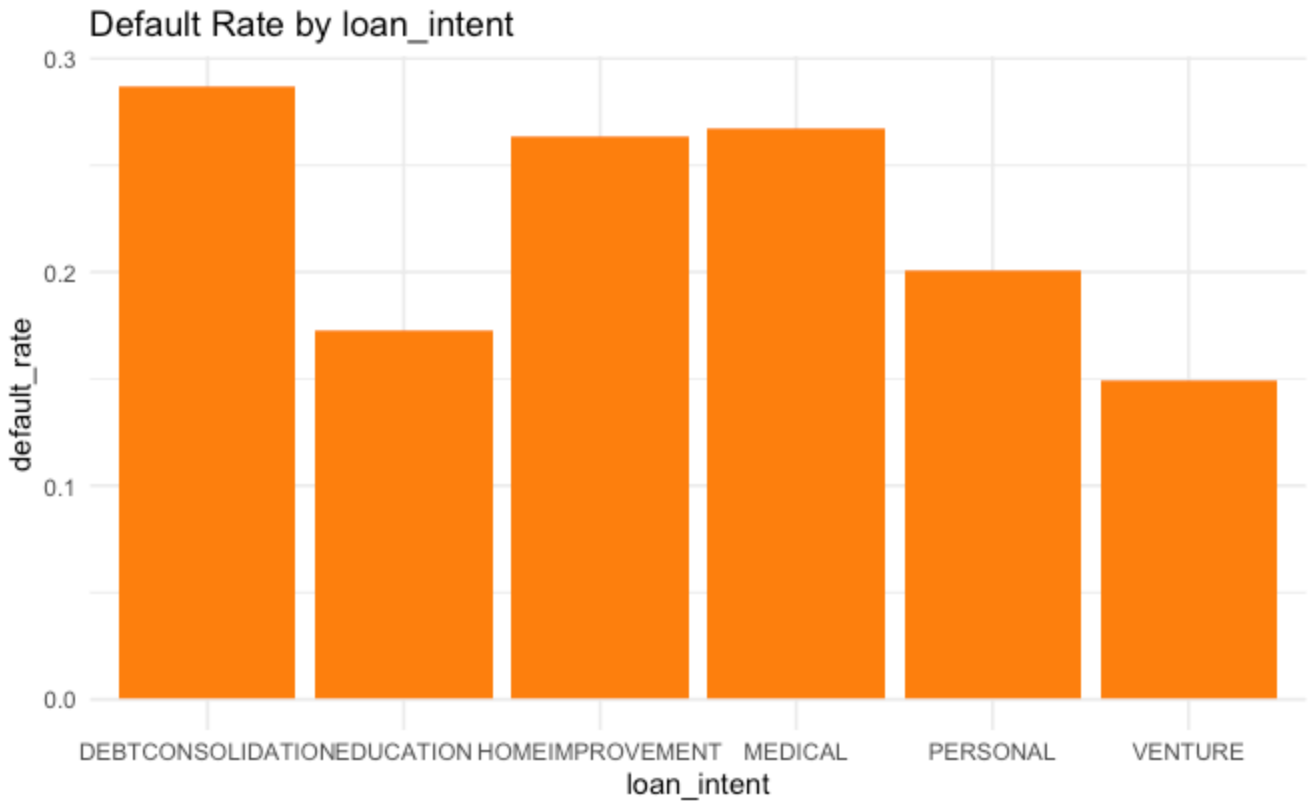


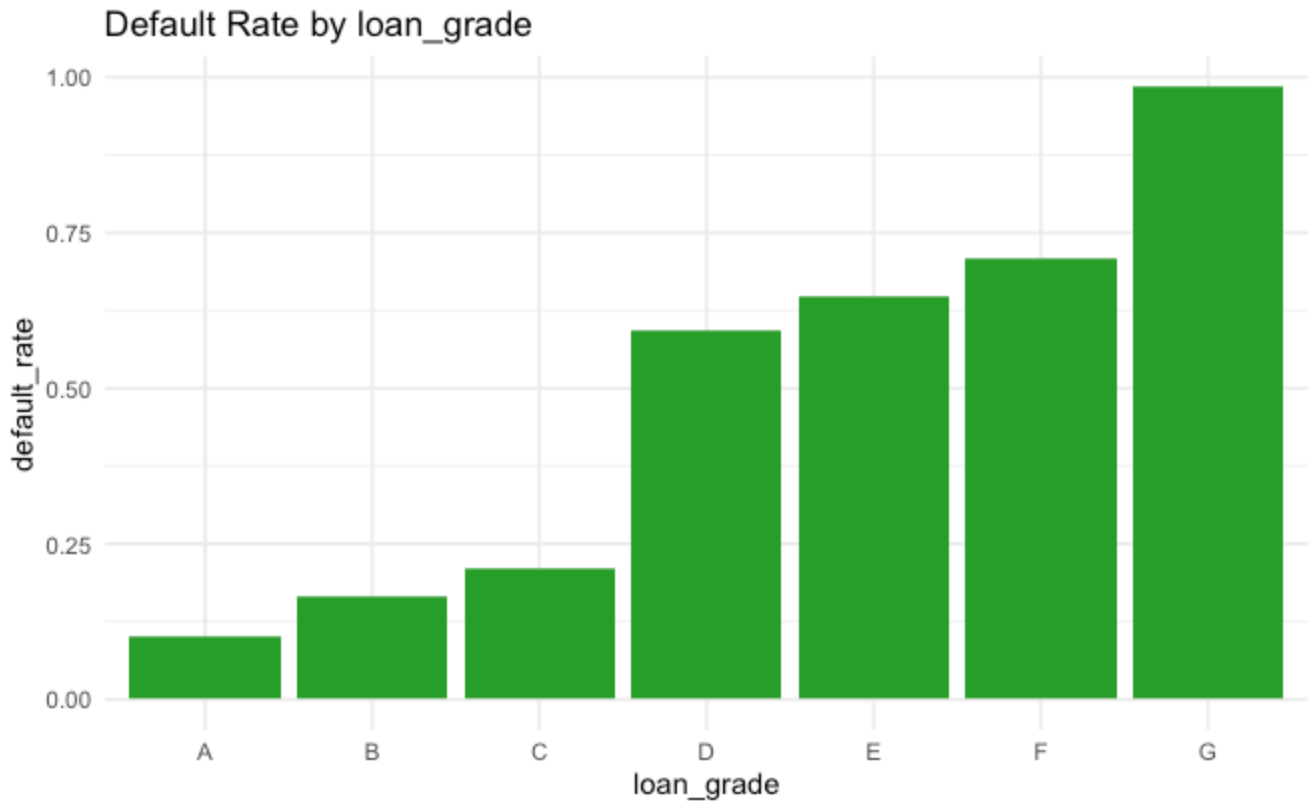
Step 5: Bivariate Analysis (Default Rates)

Hide

```
data <- data %>% mutate(loan_status_num = as.numeric(as.character(loan_status)))

rate_plot <- function(df, group_var, fill_color) {
  df %>%
    group_by(.data[[group_var]]) %>%
    summarise(default_rate = mean(loan_status_num, na.rm = TRUE), .groups = "drop") %>%
    ggplot(aes(x = .data[[group_var]], y = default_rate)) +
    geom_bar(stat = "identity", fill = fill_color) +
    ggtitle(paste("Default Rate by", group_var)) +
    theme_minimal()
}

cat_vars <- c("person_home_ownership", "loan_intent", "loan_grade")
cat_colors <- c("#1f77b4", "#ff7f0e", "#2ca02c")
walk2(cat_vars, cat_colors, ~ print(rate_plot(data, .x, .y)))
```


Default Rate by person_home_ownership
default_rate
0.0
0.1
0.2
0.3
MORTGAGE
OTHER
OWN
RENT
person_home_ownership

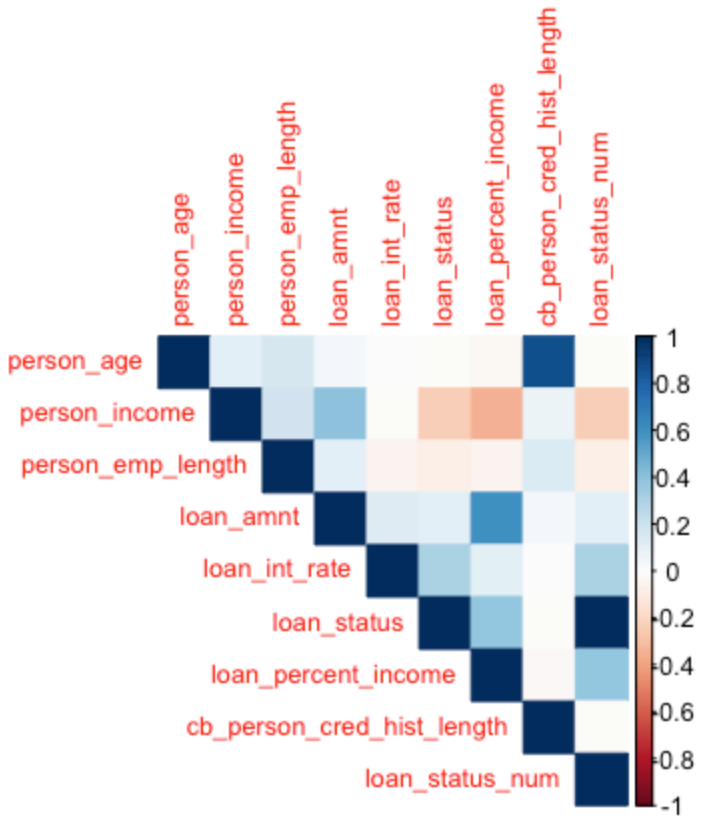

Default Rate by loan_intent
default_rate
0.0
0.1
0.2
0.3
DEBTCONSOLIDATION
EDUCATION
HOMEIMPROVEMENT
MEDICAL
PERSONAL
VENTURE
loan_intent



Step 6: Correlation Matrix

Hide

```
numeric_vars <- select(data, where(is.numeric))
corr_matrix <- cor(numeric_vars, use = "complete.obs")
corrplot(corr_matrix, method = "color", type = "upper", tl.cex = 0.8)
```



Step 7: Model Preparation

Hide

```
data <- data %>%
  mutate(
    loan_status = factor(ifelse(loan_status == "0", "NonDefault", "Default")),
    across(where(is.character), as.factor)
  ) %>%
  select(-loan_status_num) %>%
  na.omit()

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

Step 8: Logistic Regression

Hide

```
model <- train(
  loan_status ~ ., data = train_data, method = "glm", family = "binomial",
  trControl = trainControl(method = "cv", number = 5)
)

print(summary(model$finalModel))
```

```
Call:
NULL

Coefficients:
                             Estimate Std. Error z value Pr(>|z|)
(Intercept)                 3.978e+00  2.068e-01  19.235  < 2e-16 ***
person_age                  1.462e-03  6.514e-03   0.224  0.82242
person_income               1.486e-06  1.394e-06   1.066  0.28641
person_home_ownershipOTHER -5.309e-01  3.118e-01  -1.703  0.08861 .
person_home_ownershipOWN    1.746e+00  1.126e-01  15.513  < 2e-16 ***
person_home_ownershipRENT  -7.983e-01  4.507e-02 -17.713  < 2e-16 ***
person_emp_length           1.345e-02  5.388e-03   2.495  0.01258 *
loan_intentEDUCATION        8.390e-01  6.381e-02  13.148  < 2e-16 ***
loan_intentHOMEIMPROVEMENT -8.883e-02  7.125e-02  -1.247  0.21248
loan_intentMEDICAL          1.556e-01  6.018e-02   2.586  0.00971 **
loan_intentPERSONAL         6.044e-01  6.519e-02   9.271  < 2e-16 ***
loan_intentVENTURE          1.110e+00  6.968e-02  15.927  < 2e-16 ***
loan_gradeB                -2.231e-01  7.094e-02  -3.144  0.00166 **
loan_gradeC                -4.043e-01  1.011e-01  -4.000 6.34e-05 ***
loan_gradeD                -2.517e+00  1.235e-01 -20.377  < 2e-16 ***
loan_gradeE                -2.751e+00  1.612e-01 -17.062  < 2e-16 ***
loan_gradeF                -3.064e+00  2.410e-01 -12.716  < 2e-16 ***
loan_gradeG                -6.542e+00  1.037e+00  -6.306 2.87e-10 ***
loan_amnt                   1.013e-04  7.489e-06  13.527  < 2e-16 ***
loan_int_rate              -6.130e-02  1.436e-02  -4.268 1.97e-05 ***
loan_percent_income        -1.293e+01  4.095e-01 -31.572  < 2e-16 ***
cb_person_default_on_fileY -3.037e-02  5.611e-02  -0.541  0.58830
cb_person_cred_hist_length  1.487e-03  9.998e-03   0.149  0.88176
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 27133  on 25800  degrees of freedom
Residual deviance: 17547  on 25778  degrees of freedom
AIC: 17593

Number of Fisher Scoring iterations: 6
```

Hide

```
predictions <- predict(model, test_data)
conf_matrix <- confusionMatrix(predictions, test_data$loan_status)
print(conf_matrix)
```

```
Confusion Matrix and Statistics

            Reference
Prediction   Default NonDefault
  Default        794        245
  NonDefault     619       4791

               Accuracy : 0.866
                 95% CI : (0.8575, 0.8742)
    No Information Rate : 0.7809
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.5673

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.5619
            Specificity : 0.9514
         Pos Pred Value : 0.7642
         Neg Pred Value : 0.8856
             Prevalence : 0.2191
         Detection Rate : 0.1231
   Detection Prevalence : 0.1611
      Balanced Accuracy : 0.7566

       'Positive' Class : Default
```

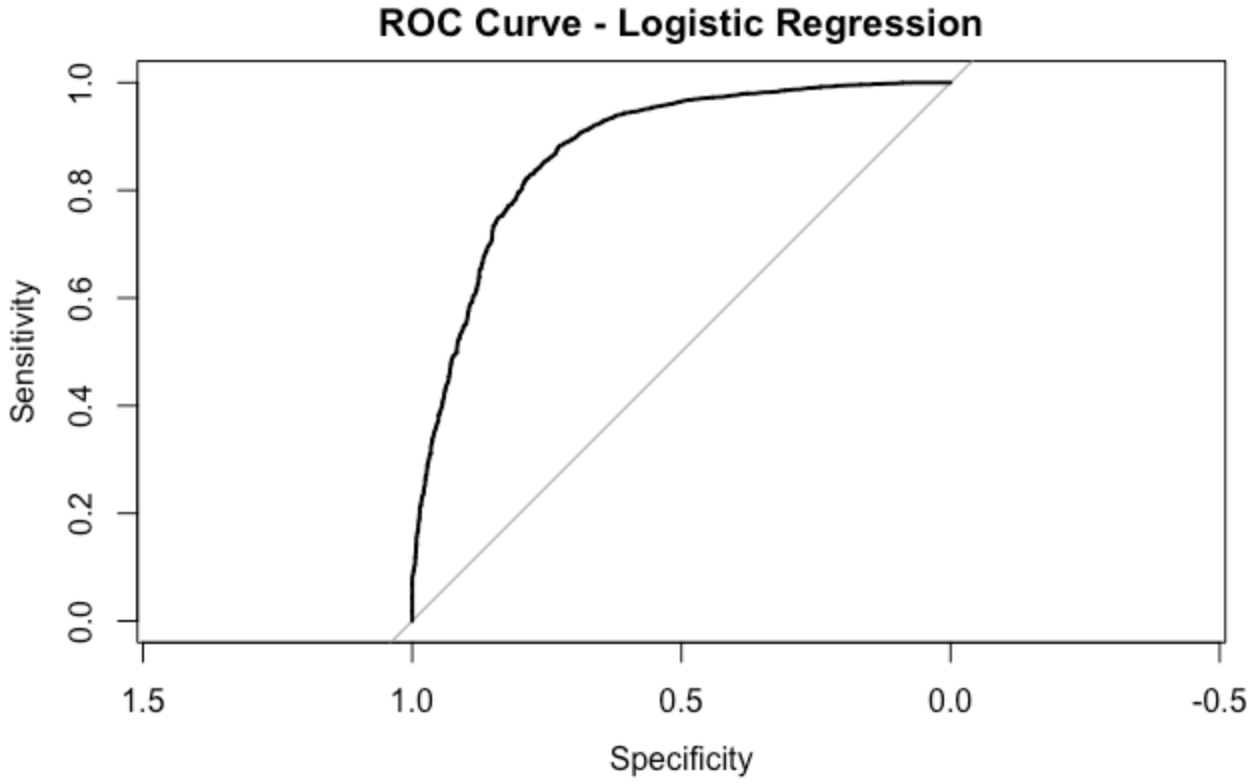
Hide

```
# ROC and AUC
prob_predictions <- predict(model, test_data, type = "prob")[, 2]
roc_obj <- roc(as.numeric(test_data$loan_status) - 1, prob_predictions)
```

```
Setting levels: control = 0, case = 1
Setting direction: controls < cases
```

Hide

```
plot(roc_obj, main = "ROC Curve - Logistic Regression")
```



Hide

```
cat("AUC:", auc(roc_obj), "\n")
```

```
AUC: 0.8741455
```

Step 9: PCA + Logistic Regression

Hide

```
numeric_train <- select(train_data, where(is.numeric)) %>% mutate_all(as.numeric)
preproc <- preProcess(numeric_train, method = c("center", "scale"))
numeric_train_scaled <- predict(preproc, numeric_train)

pca_result <- prcomp(numeric_train_scaled)
explained_var <- summary(pca_result)$importance["Cumulative Proportion", ]
num_pcs <- which(explained_var >= 0.95)[1]
cat("Number of PCs kept:", num_pcs, "\n")
```

```
Number of PCs kept: 5
```

Hide

```
# Train data with selected PCs
train_pcs <- as.data.frame(pca_result$x[, 1:num_pcs])
colnames(train_pcs) <- paste0("PC", 1:num_pcs)
train_data_pca <- bind_cols(train_data, train_pcs)

# Test data
numeric_test_scaled <- predict(preproc, select(test_data, where(is.numeric)) %>% mutate_
all(as.numeric))
test_pca_scores <- as.data.frame(as.matrix(numeric_test_scaled) %*% pca_result$rotation
[, 1:num_pcs])
colnames(test_pca_scores) <- paste0("PC", 1:num_pcs)
test_data_pca <- bind_cols(test_data, test_pca_scores)

# Logistic regression on PCs
pca_model <- train(
  as.formula(paste("loan_status ~", paste0("PC", 1:num_pcs, collapse = " + "))),
  data = train_data_pca,
  method = "glm", family = "binomial",
  trControl = trainControl(method = "cv", number = 5)
)

pca_predictions <- predict(pca_model, test_data_pca)
print(confusionMatrix(pca_predictions, test_data_pca$loan_status))
```

```
Confusion Matrix and Statistics

            Reference
Prediction   Default NonDefault
  Default        576        263
  NonDefault     837       4773

               Accuracy : 0.8294
                 95% CI : (0.82, 0.8385)
    No Information Rate : 0.7809
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.4162

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.40764
            Specificity : 0.94778
         Pos Pred Value : 0.68653
         Neg Pred Value : 0.85080
             Prevalence : 0.21910
         Detection Rate : 0.08932
   Detection Prevalence : 0.13010
      Balanced Accuracy : 0.67771

       'Positive' Class : Default
```

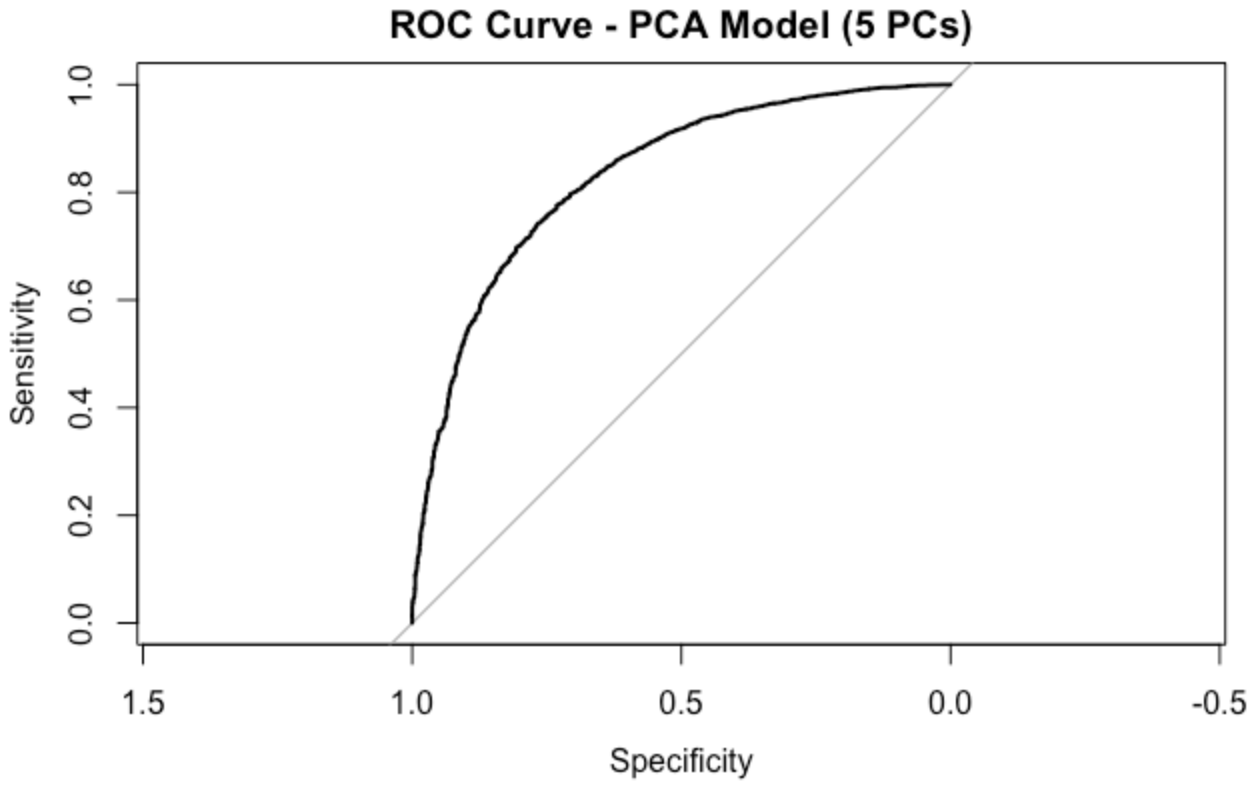
Hide

```
pca_prob_predictions <- predict(pca_model, test_data_pca, type = "prob")[, 2]
pca_roc_obj <- roc(as.numeric(test_data_pca$loan_status) - 1, pca_prob_predictions)
```

```
Setting levels: control = 0, case = 1
Setting direction: controls < cases
```

Hide

```
plot(pca_roc_obj, main = paste("ROC Curve - PCA Model (", num_pcs, " PCs)", sep = ""))
```



Hide

```
cat("PCA AUC:", auc(pca_roc_obj), "\n")
```

```
PCA AUC: 0.8304693
```

Step 10: Compare Model AUCs

Hide

```
cat("AUC:", auc(roc_obj), "\n")
```

```
AUC: 0.8741455
```

Hide

```
cat("PCA AUC:", auc(pca_roc_obj), "\n")
```

```
PCA AUC: 0.8304693
```