Final Project Data Mining

Borrower behavior and loan outcome correlation

Jasa Gellert

Introduction

The current climate is highly competitive, including the lending environment with banks. Effective risk management and choosing consumers to allocate loans is important for any bank looking to maintaining profitability and long-term stability. We understand your bank is facing challenges related to loan defaults and loan portfolio optimization. Using Data Analysis in R Studio, summary tables, data visualization, and machine learning tools, we can help the bank understand which borrower demographics pose higher financial risk, and which characteristics lead to better loan performance.

To find patterns in borrower behavior and loan outcomes, we'll answer several critical business questions. Specifically, we examined the relationship between loan defaults and variables such as loan amount, credit lines, homeownership status, job tenure, and application type (individual vs. joint). Summary tables and graphs were used to visualize the data and identify trends, which we can then analyze.

Larger loan amounts had a stronger correlation with defaults than small loan amounts. The $15,000–$30,000 range was the most prominent. Over 52% of defaulters had loans above $15,000, suggesting the need for stricter screening or risk pricing for middle and high loan amounts. Borrowers with extremely high or low total credit lines also exhibited greater variability in default risk. Mortgage holders qualified for significantly larger loans than renters, indicating stronger financial capacity and supporting a strategy to offer higher ceilings to homeowners. Longer job tenure appeared to be a positive signal of borrower reliability and financial stability. Meanwhile, joint applicants, although receiving larger loans on average, showed higher default rates (about 45%) compared to individual applicants (about 36%).

In addition to descriptive insights, we implemented a K-Nearest Neighbors (KNN) classification model, which will predict the probability of loan default for new applicants based on historical patterns. The model helps further our findings by scoring applicants using multiple risk factors at the same time. By testing many operations, we found a version for the bank that delivers strong predictive performance. With an ROC AUC score of 0.868, analyzing this can support more accurate risk-based decision-making that will benefit your bank.

By combining traditional data analysis with predictive modeling, this project aims to improve efficiency, reduce default rates, and tailor loan products to different customer segments. The KNN model will allow the bank to be proactive, identifying at-risk borrowers

before they can receive a loan from the bank, and optimizing the portfolio for both risk and growth.

Data Analysis

Question 1

What is the correlation between loan_default and loan_amount?

Answer: Based on the loan data provided, there is a moderate relationship between loan amount and default rates. Applicants who defaulted on their loans ("yes") had a higher average loan amount ($17,447.53) compared to those who did not default ($16,245.21). Additionally, a higher percentage of defaulting applicants (52.2%) took out loans over $15,000, compared to 42.7% among non-defaulters. This pattern suggests that larger loan amounts may be associated with increased financial risk, possibly due to those needing to repay them, which causes a lot of mental and financial strain. While not a direct measure of correlation (like Pearson's r), these distribution differences strongly imply that loan size is a relevant factor in predicting default.

Further insights come from examining the percentage of loans above higher thresh holds. Among defaulters, 33% had loans over $20,000, compared to 28% of non-defaulters. Interestingly, this trend reverses at the $30,000 threshold. Only 8.6% of defaulters had loans above $30,000, compared to 11.5% of non-defaulters. This may indicate that the top highest loan recipients are more carefully screened or have stronger financial profiles, whereas the mid-high range of $15,000–$30,000 might represent that borrowers are more likely to struggle with repayment. It may also reflect riskier lending practices for mid-range loans where the criteria for taking out bank fees are not as harsh.

For the bank to reduce default rates, our recommendation is that the company considers being stricter on approval criteria for loans between $15,000 and $30,000. You may do this by requiring higher credit scores, more documentation, or having to co-sign. Additionally, adding tiered risk assessments or interest rate adjustments based on loan size and risk profile could help manage exposure. Educating borrowers about the long-term financial commitment of mid-sized loans and offering financial counseling or pre-approval assessments could further help reduce defaults. To conclude, matching loan amounts more closely with borrower capacity, especially in the $15,000–$30,000 range, appears to be the best strategy for reducing defaults.

Question 2

What is the correlation between loan_default and total_credit_lines?

Answer: The violin plot illustrating the relationship between loan default and total credit lines reveals important trends. Both defaulters and non-defaulters have a similar score in credit line counts, with most individuals concentrated around the 10–30 range. However,

the distribution among defaulters appears slightly wider and flatter, which indicates more differences between, and a good amount of borrowers with either very low or very high credit lines. The central tendency is comparable, but the spread suggests that extreme credit line values could be a risk factor for loan default. We can conclude that individuals with either limited or overly extended credit access may be more likely to default.

One key recommendation is to implement stricter lending criteria for applicants with very high or very low total credit lines. For example, borrowers with fewer than 5 or more than 50 credit lines could be flagged for additional financial review or risk assessment. The broader spread in the violin plot among defaulters is good evidence, and it suggests that customers on the edge of the violin graph may carry greater risk. Increasing screening for these applicants will help the company filter out high-risk profiles, and not have to deny access to the stable middle range of borrowers.

Since the bank can now focus on the most risky credit line patterns, the company can reduce overall loan defaults, protecting its loan portfolio and improving profitability. Fewer defaults lead to lower collection costs, reduced write-offs, and higher recovery rates. Additionally, this targeted adjustment avoids penalizing average borrowers and enables the business to maintain loan volume while improving quality. This is a strategy using data and risk to your advantage, which also enhances financial performance without limiting potential growth.

Question 3

Who has the higher loan_amount, rent, mortgage, or own (homeownership variable)?

Answer: Loan amounts across different homeownership roles shows a clear distinction in borrowing behavior, especially between renters and those with a mortgage. According to the data, individuals with a mortgage have the highest mean loan amount at $18,198.61, compared to renters' loans of $14,996.46. Those who have a home are at $16,513.95. Having a difference of over $3,000 between renters and mortgage owners shows that borrowers with mortgages are more likely to qualify for and receive larger loans. Using the boxplot as our evidence, mortgage holders have a higher distribution, with median and upper quartiles exceeding renters numbers.

Our recommendation is that the company evaluates loan applications from renters, as they generally receive smaller loan amounts and may present a different risk profile. Renters may lack the financial assets or stability that often come with homeownership, which has the potential to affect their repayment ability. This doesn't mean that renters are higher risk by default, though. A lower borrowing amounts could reflect underlying financial constraints, which should still be accounted for. By offering special loans or lower limits to renters, the bank could help reduce exposure while still serving the demographic.

By offering custom loan terms for renters vs. mortgage holders, the company can reduce default risks while better aligning loan offers with borrower capacity. This strategy

promotes responsible lending, enhances customer satisfaction through tailored offers, and protects the company's loan portfolio. Moreover, directing more favorable terms or higher loan ceilings toward mortgage holders, who show higher loan averages, can improve approval rates and drive revenue from safer lending segments.

Question 4

Do longer job lengths (current_job_years) reduce loan amounts (loan_amount)?

Answer: The Pearson correlation coefficient between current job years and loan amount is approximately 0.10, with a p-value of 1.29e-10. This indicates a weak but statistically significant positive correlation, which means that while longer job tenure is slightly associated with higher loan amounts, the strength of this relationship is limited. The 95% confidence interval (0.07 to 0.13) reinforces the reliability of this small but consistent positive trend. So in regards to the question, longer job lengths do not reduce loan amounts. They're actually linked with modestly higher ones.

The scatterplot with a fitted regression line visually confirms the statistical findings: as current job years increase, there is a slight upward trend in loan amounts. While the points are widely scattered—indicating high variability—the blue trend line shows a gentle incline, matching the weak positive correlation coefficient ($r \approx 0.10$). This suggests that longer job tenure is loosely associated with slightly larger loan amounts, though other factors likely play a much larger role.

Further support comes from the grouped summary by job_bin. Applicants with 0–2 years at their current job have a mean loan of $15,584, while those with 6–10 years have a significantly higher average of $17,587. Median values follow the same upward trend. These results show that applicants with longer employment histories tend to qualify for or request larger loan amounts, possibly due to more stable income, improved creditworthiness, or increased lender trust.

Given this, the company should consider leveraging job tenure as a factor in loan evaluations. Borrowers with longer employment history appear to qualify for larger loans and may also represent more stable lending candidates. Incorporating job length into credit models or approval criteria could improve risk prediction, helping the company offer higher-value loans to reliable customers while potentially lowering default rates. This strategy can boost portfolio quality and customer lifetime value, making it both a financially and operationally sound business decision.

For the business, this means incorporating job tenure as a soft risk indicator can lead to smarter lending decisions. Applicants with just 0–2 years at their current job average lower loans, while those with 6–10 years average over $17,500. A policy that rewards stable employment (higher approved limits or faster approvals) can encourage applications from more financially stable borrowers and help reduce default rates—protecting profits, while improving portfolio health.

Question 5

How many people apply for loans (loan_default) individually compared to joint? (application_type)

Answer: Based on the data, a significantly higher number of people apply for loans individually rather than jointly. Specifically, out of all applicants, 3,494 applied individually (1,253 defaulted and 2,241 did not), while only 616 applied jointly (277 defaulted and 339 did not). This shows that individual applications are the dominant mode, comprising over 85% of the total.

However, when examining default rates by application type, joint applications show a higher proportion of defaults. About 44.97% of joint applicants defaulted, compared to 35.86% of individual applicants. This suggests that joint applications have a higher relative risk of default despite being fewer in number.

The company should closely monitor and possibly reassess its risk assessment model for joint applications. Since these applicants default at a higher rate proportionally, it may be beneficial to implement stricter underwriting criteria for joint loans or require additional documentation that evaluates the combined financial health of both applicants. Doing so could help lower the default rate, reduce financial losses, and strengthen the loan portfolio. Additionally, improved screening may lead to more sustainable joint loans, which can still be valuable for high-amount lending.

Question 6

Based on the data, does it seem better financially to apply individually or joint? (loan_amount and application_type)

Answer: Based on the data, it appears more financially advantageous to apply jointly rather than individually when considering the loan amount received. Applicants who applied jointly were approved for loans with a mean value of $20,622.48 and a median of $19,750, compared to $15,999.98 mean and $14,400 median for individual applicants. This indicates that joint applicants, on average, qualify for higher loan amounts.

Despite higher default proportions among joint applicants (as noted in earlier analysis), the financial benefit of a larger approved loan may outweigh the added risk for some borrowers. The standard deviation is also slightly higher for joint applications, suggesting more variability, but this is expected given the typically larger financial profiles involved in joint applications.

If the business aims to maximize approved loan volumes, encouraging joint applications could be beneficial, especially for borrowers seeking larger loans. However, due to their higher default rate, joint applications should be subject to enhanced credit checks or dual-income validation procedures. By combining this approach with careful risk segmentation, the company can increase loan revenue while controlling potential losses, ultimately improving profitability.

Machine Learning

To help the bank better identify applicants at risk of defaulting on their loans, we developed and evaluated a classification model using the k-nearest neighbors (KNN) algorithm. Several versions of the KNN model were tested, each using a different number of "neighbors", which is a parameter that controls how the model classifies a new applicant based on past similar cases. After comparing the results, we found that the model with 15 neighbors delivered the best performance overall.

A key measure we used to evaluate model effectiveness is the ROC AUC score, which tells us how well the model separates high-risk applicants from low-risk ones. The ROC AUC for the best model was 0.868, indicating a strong ability to distinguish between applicants who will and will not default. In simple terms, the closer this score is to 1, the better the model is at ranking applicants by their likelihood of default. A score of 0.868 suggests the model is very effective for this purpose.

To estimate how this model might perform on future, unseen loan applications, we tested it using part of our data that was held out for validation. The confusion matrix from this test revealed that the model correctly identified 245 customers who did default and 589 customers who did not, while only making a small number of classification mistakes. This gives us confidence that the model can generalize well and support better decision-making.

Enhance Risk Screening During Application Review:

The model can be used to flag applicants who are most likely to default. These flagged applicants should undergo more rigorous review, including possible requirements for a co-signer, lower loan limits, or additional financial documentation.

Tailor Loan Policies Based on Application Type:

Our exploratory analysis (including boxplots of loan amounts by application type) revealed that some application types are consistently associated with higher loan amounts, which may correlate with increased risk. Adjusting loan approval criteria or limits for higher-risk application types can help mitigate potential losses.

#Increase Monitoring of Moderate-Risk Borrowers:

Applicants who fall in a mid-risk category, in between clear high-risk and low-risk, should receive targeted support. This could include financial counseling from the bank, proactive outreach, or more frequent online, mail, or text payment reminders to ensure they stay on track with repayment.

By using this predictive model, the bank can improve its risk management strategy, and make more informed lending decisions, which ultimately reduces financial losses due to

loan defaults. The KNN model provides a great opportunity to enhance loan portfolio performance, and support customer access to bank loans.

Conclusion

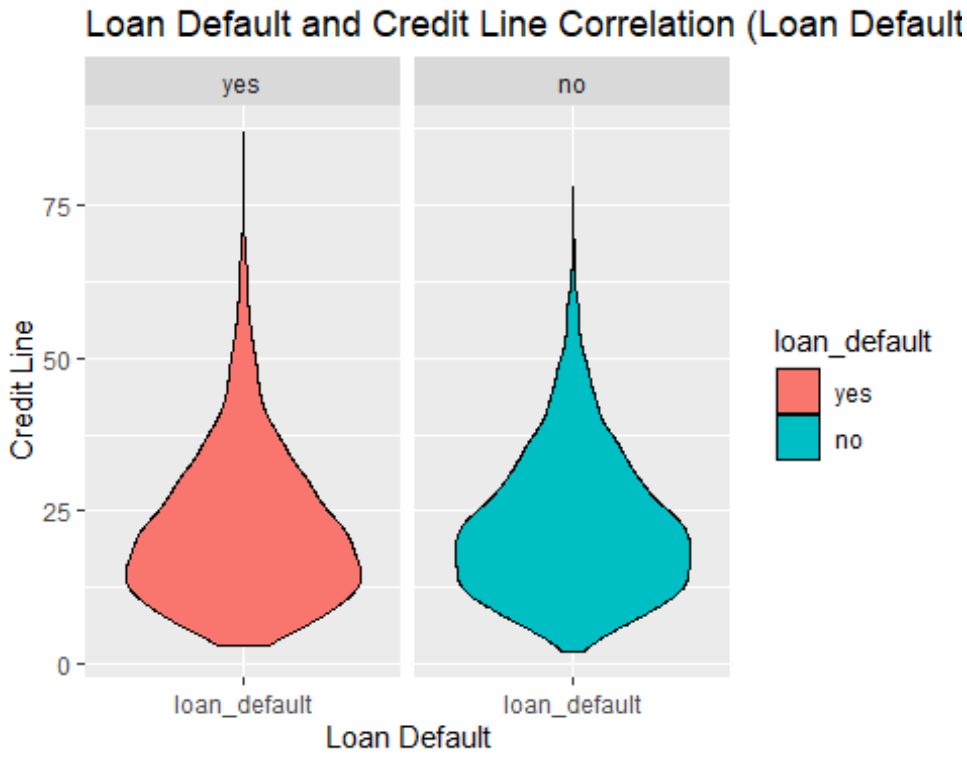
This well-rounded and detailed analysis has demonstrated how data visualization, interpreting data, and machine learning can work in tandem to improve loan risk assessment. Key variables such as loan amount, total credit lines, homeownership status, job tenure, and application type were all found to meaningfully influence loan default risk. Patterns uncovered in the data support actionable recommendations, such as stricter screening for mid-sized loans, offering larger loans to mortgage holders, using job tenure as a proxy for financial stability, and applying targeted policies for joint applicants. These insights help not only in understanding borrower behavior but also in identifying specific risk factors that can be addressed to optimize the bank's lending practices.

More importantly, the integration of a K-Nearest Neighbors (KNN) model brings predictive power to these insights. With a strong ROC AUC score of 0.868, the model offers reliable forecasting of default risk based on applicant profiles. Now your bank can identify high-risk borrowers before loans are issued. The model can flag applicants with demographics who are more likely to fail loan payments, and be cautious when issuing them out. By using both statistical analysis and machine learning into the loan evaluation process, your bank can lower default rates, improve quality for stakeholders and the business, and continue to grow in a competitive banking industry.

Appendix R codes and data visualization plus machine learning

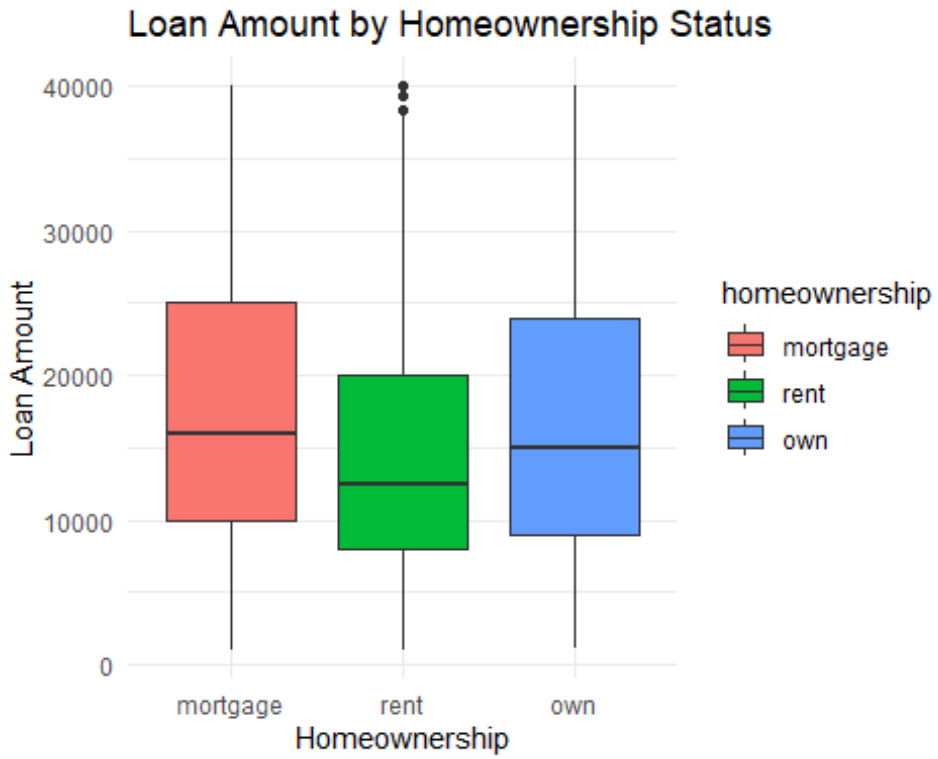
```r
# Suppress dplyr summarise grouping warning messages
options(dplyr.summarise.inform = FALSE)

## Add R libraries here
library(tidyverse)
library(tidymodels)

# Load data
loan_data <- readRDS("C:/Users/15713/Downloads/loan_data.rds")

# Question 1 What is the correlation between loan_default and loan_amount?
# Summary table groups loan_data by loan_default
# Finds min, max, avg, standard deviation, and percentages

loan_data %>% group_by(loan_default) %>%
                  summarise(n_loan_amount = n(),
                            min_loan_amount = min(loan_amount),
                            max_loan_amount = max(loan_amount),
                            avg_loan_amount = mean(loan_amount),
                            sd_loan_amount = sd(loan_amount),
                            pct_over_15000 = mean(loan_amount > 15000),
```

```
                            pct_over_20000 = mean(loan_amount > 20000),
                            pct_over_30000 = mean(loan_amount > 30000))

## # A tibble: 2 × 9
##   loan_default n_loan_amount min_loan_amount max_loan_amount
avg_loan_amount
##   <fct>                <int>           <int>           <int>
<dbl>
## 1 yes                   1530            1000           40000
17448.
## 2 no                    2580            1000           40000
16245.
## # ℹ 4 more variables: sd_loan_amount <dbl>, pct_over_15000 <dbl>,
## #   pct_over_20000 <dbl>, pct_over_30000 <dbl>

# Question 2 What is the correlation between loan_default and
total_credit_lines?

# Summarize the data
summary_data <- loan_data %>%
  group_by(loan_default) %>%
  summarise(avg_credit_lines = mean(total_credit_lines, na.rm = TRUE))

# Create Violin plot which will show discrepencies

ggplot(data = loan_data, aes(x = 'loan_default', fill = loan_default)) +
   geom_violin(aes(y = total_credit_lines), color = alpha('black', 5)) +
   facet_wrap(~ loan_default) +
   labs(title = "Loan Default and Credit Line Correlation (Loan Default -
Yes/No)",
         x = "Loan Default", y = "Credit Line")
```

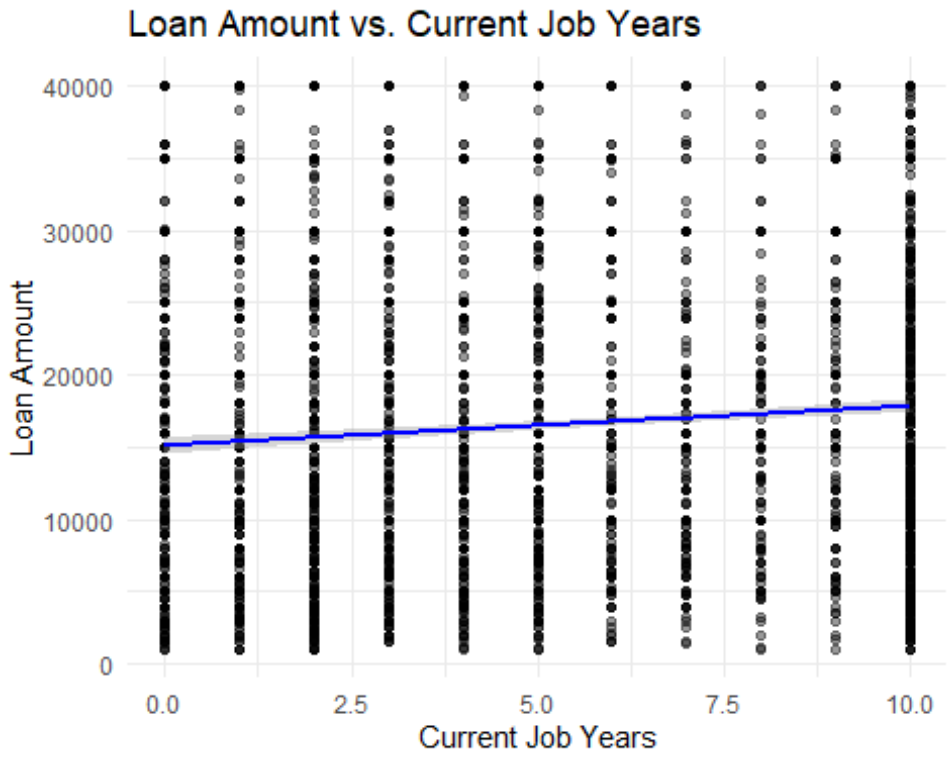



```
# Question 3 Who has the higher loan_amount, rent, mortgage, or own
(homeownership variable)?
# Summary table groups data by homeownership
# Counts mean, median, standard deviation, min, max

loan_data %>%
  group_by(homeownership) %>%
  summarise(
    count = n(),
    mean_loan_amount = mean(loan_amount, na.rm = TRUE),
    median_loan_amount = median(loan_amount, na.rm = TRUE),
    sd_loan_amount = sd(loan_amount, na.rm = TRUE),
    min_loan_amount = min(loan_amount, na.rm = TRUE),
    max_loan_amount = max(loan_amount, na.rm = TRUE)
  )

## # A tibble: 3 × 7
##   homeownership count mean_loan_amount median_loan_amount sd_loan_amount
##   <fct>         <int>            <dbl>              <dbl>          <dbl>
## 1 mortgage       1937           18199.              16000         10244.
## 2 rent           1666           14996.              12500          9546.
## 3 own             507           16514.              15000          9941.
## # ℹ 2 more variables: min_loan_amount <int>, max_loan_amount <int>

ggplot(loan_data, aes(x = homeownership, y = loan_amount, fill =
homeownership)) +
  geom_boxplot() +
```

```
  labs(
    title = "Loan Amount by Homeownership Status",
    x = "Homeownership",
    y = "Loan Amount"
  ) +
  theme_minimal()
```




```
# Question 4

ggplot(loan_data, aes(x = current_job_years, y = loan_amount)) +
  geom_point(alpha = 0.4) +
  geom_smooth(method = "lm", se = TRUE, color = "blue") +
  labs(
    title = "Loan Amount vs. Current Job Years",
    x = "Current Job Years",
    y = "Loan Amount"
  ) +
  theme_minimal()
```




```r
cor.test(loan_data$current_job_years, loan_data$loan_amount, use =
"complete.obs")

##
##  Pearson's product-moment correlation
##
## data:  loan_data$current_job_years and loan_data$loan_amount
## t = 6.4454, df = 4108, p-value = 1.286e-10
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.06969643 0.13023255
## sample estimates:
##       cor
## 0.1000571

library(dplyr)

loan_data %>%
  mutate(job_bin = cut(current_job_years, breaks = c(0, 2, 5, 10, 20, Inf),
                       labels = c("0-2", "3-5", "6-10", "11-20", "20+"))) %>%
  group_by(job_bin) %>%
  summarise(
    mean_loan = mean(loan_amount, na.rm = TRUE),
    median_loan = median(loan_amount, na.rm = TRUE),
    count = n()
  )
```

```
## # A tibble: 4 × 4
##   job_bin mean_loan median_loan count
##   <fct>       <dbl>       <dbl> <int>
## 1 0-2        15584.       12900   772
## 2 3-5        16236.       14500   990
## 3 6-10       17588.       15300  2032
## 4 <NA>       15077.       12900   316
```

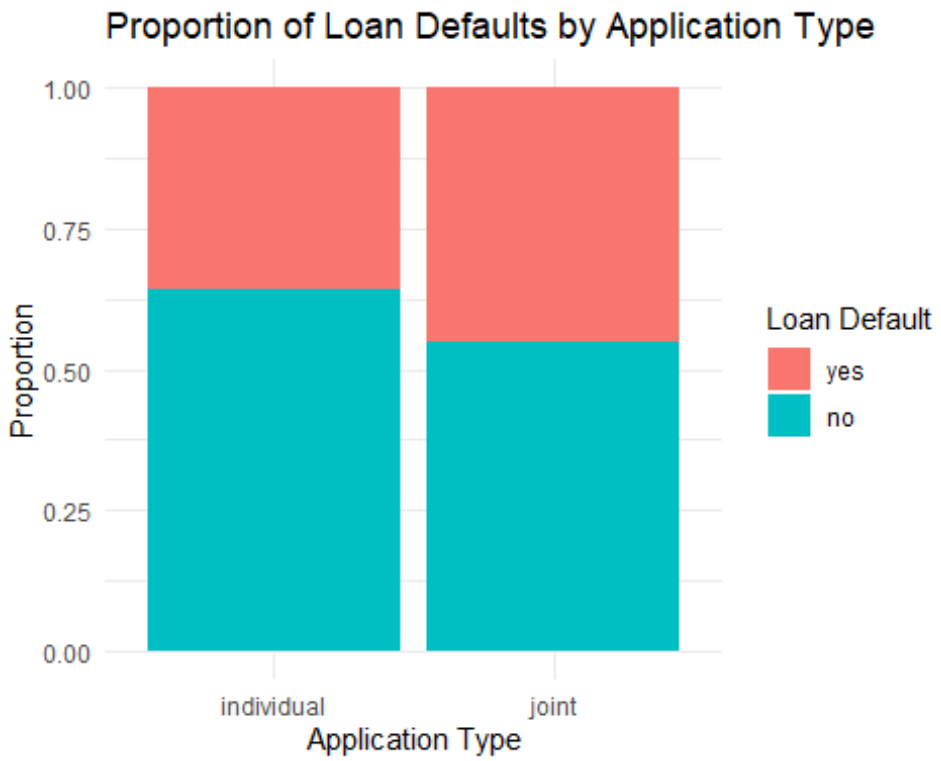
```
library(ggplot2)

# Create a contingency table
table_summary <- table(loan_data$application_type, loan_data$loan_default)
print(table_summary)
```

```
##
##                yes   no
##   individual  1253 2241
##   joint        277  339
```

```
# Proportional table
prop.table(table_summary, margin = 1)  # Proportions within each
application_type
```

```
##
##                     yes        no
##   individual  0.3586148 0.6413852
##   joint       0.4496753 0.5503247
```

```
ggplot(loan_data, aes(x = application_type, fill = loan_default)) +
  geom_bar(position = "fill") +  # Shows proportions
  labs(title = "Proportion of Loan Defaults by Application Type",
       x = "Application Type",
       y = "Proportion",
       fill = "Loan Default") +
  theme_minimal()
```

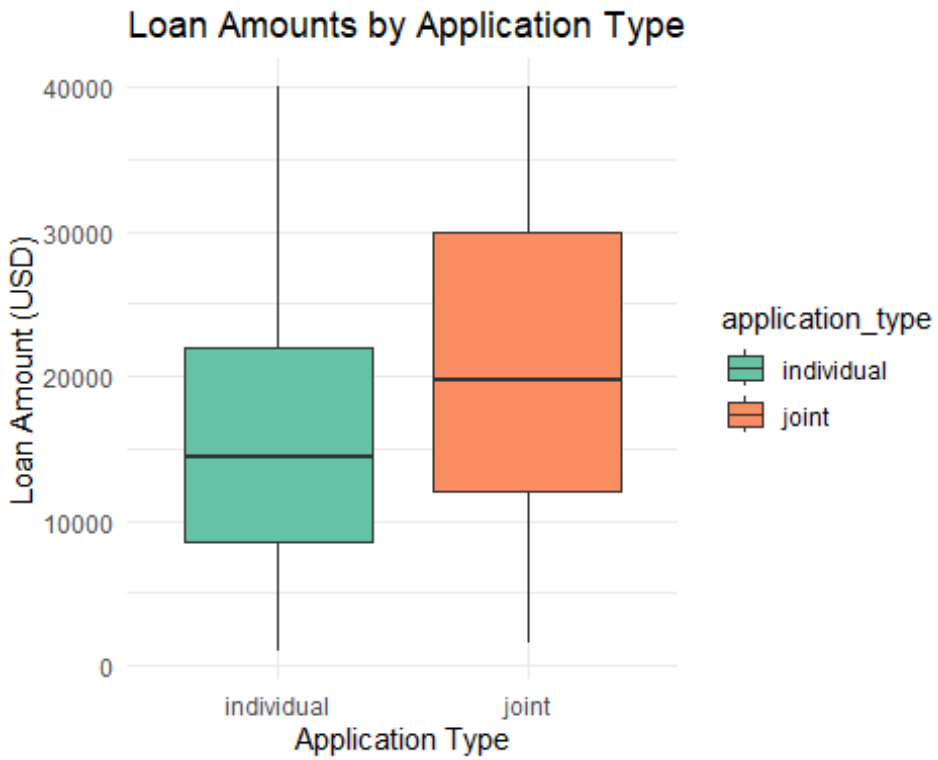



```r
# Load dplyr for summarization
library(dplyr)

application_loan <- loan_data %>%
  group_by(application_type) %>%
  summarise(
    count = n(),
    mean_loan = mean(loan_amount, na.rm = TRUE),
    median_loan = median(loan_amount, na.rm = TRUE),
    sd_loan = sd(loan_amount, na.rm = TRUE),
    min_loan = min(loan_amount, na.rm = TRUE),
    max_loan = max(loan_amount, na.rm = TRUE)
  )

print(application_loan)
```

```
## # A tibble: 2 × 7
##   application_type count mean_loan median_loan sd_loan min_loan max_loan
##   <fct>            <int>     <dbl>       <dbl>   <dbl>    <int>    <int>
## 1 individual        3494    16000.       14400   9768.     1000    40000
## 2 joint              616    20622.       19750  10643.     1500    40000
```

```r
library(ggplot2)

ggplot(loan_data, aes(x = application_type, y = loan_amount, fill =
application_type)) +
  geom_boxplot() +
```

```
  labs(title = "Loan Amounts by Application Type",
       x = "Application Type",
       y = "Loan Amount (USD)") +
  theme_minimal() +
  scale_fill_brewer(palette = "Set2")
```



Classification Model and Machine Learning

Model 1

```
suppressPackageStartupMessages(library(tidyverse))
suppressPackageStartupMessages(library(tidymodels))
suppressPackageStartupMessages(library(discrim))

## Warning: package 'discrim' was built under R version 4.4.3

suppressPackageStartupMessages(library(klaR))

## Warning: package 'klaR' was built under R version 4.4.3

suppressPackageStartupMessages(library(kknn))

## Warning: package 'kknn' was built under R version 4.4.3

loan_data <- readRDS("C:/Users/15713/Downloads/loan_data.rds")

set.seed(123)
```

```r
loan_split <- initial_split(loan_data, strata = loan_default)

loan_training <- loan_split %>% training()

loan_test <- loan_split %>% testing()

loan_folds <- vfold_cv(loan_training, v = 5, strata = loan_default)
```

Model 2

```r
loan_recipe <- recipe(loan_default~., data = loan_training) %>%
                  step_YeoJohnson(all_numeric(), -all_outcomes()) %>%
                   step_normalize(all_numeric(), -all_outcomes()) %>%
                   step_dummy(all_nominal(), -all_outcomes())

loan_recipe %>%
 prep() %>%
  bake(new_data = loan_training)
```

```
## # A tibble: 3,082 × 20
##    loan_amount installment interest_rate annual_income current_job_years
##          <dbl>       <dbl>         <dbl>         <dbl>              <dbl>
##  1     1.14         1.40         -0.570          1.89               1.09
##  2    -1.62        -1.70         -0.362          0.167             -0.406
##  3    -1.24        -1.23         -0.861          0.112             -0.406
##  4     1.86         1.43         -0.0309         0.112             -0.698
##  5    -0.00630      0.0461        0.0333        -0.833              1.09
##  6     1.62         0.977        -1.09           2.23               1.09
##  7    -1.16        -1.11          0.0333        -1.35               1.09
##  8    -0.572       -0.934         0.524         -1.69              -1.35
##  9    -0.388       -0.850        -0.936         -1.17              -0.131
## 10    -0.572       -0.557        -1.09          -0.184              1.09
## # ℹ 3,072 more rows
## # ℹ 15 more variables: debt_to_income <dbl>, total_credit_lines <dbl>,
## #   years_credit_history <dbl>, loan_default <fct>,
## #   loan_purpose_credit_card <dbl>, loan_purpose_medical <dbl>,
## #   loan_purpose_small_business <dbl>, loan_purpose_home_improvement
<dbl>,
## #   application_type_joint <dbl>, term_five_year <dbl>,
## #   homeownership_rent <dbl>, homeownership_own <dbl>, …
```

```r
logistic_model <- logistic_reg() %>%
                  set_engine("glm") %>%
                  set_mode("classification")
logistic_model
```

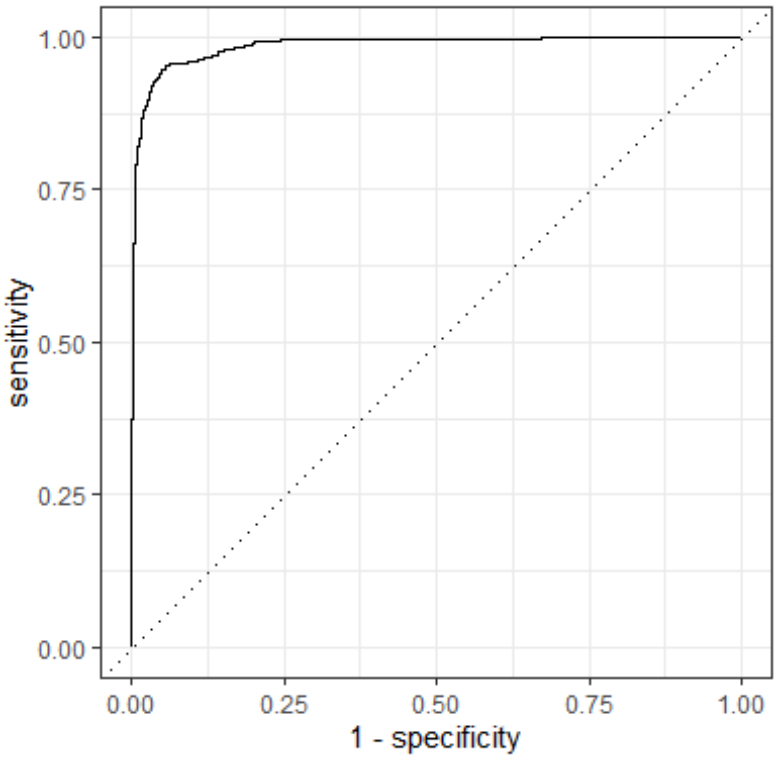
```
## Logistic Regression Model Specification (classification)
##
## Computational engine: glm
```

```r
logistic_wf <- workflow() %>%
               add_model(logistic_model) %>%
               add_recipe(loan_recipe)

logistic_fit <-   logistic_wf %>%
                last_fit(split = loan_split)

logistic_results <-  logistic_fit %>%
                     collect_predictions()

logistic_results
```

```
## # A tibble: 1,028 × 7
##    .pred_class .pred_yes .pred_no id                .row loan_default
.config
##    <fct>           <dbl>    <dbl> <chr>            <int> <fct>
<chr>
##  1 no            0.0189    0.981  train/test split     8 no
Preproces…
##  2 yes           0.891     0.109  train/test split    15 yes
Preproces…
##  3 yes           0.677     0.323  train/test split    21 yes
Preproces…
##  4 yes           0.801     0.199  train/test split    25 yes
Preproces…
##  5 no            0.104     0.896  train/test split    28 no
Preproces…
##  6 yes           0.985     0.0148 train/test split    32 yes
Preproces…
##  7 no            0.00493   0.995  train/test split    38 no
Preproces…
##  8 no            0.00750   0.992  train/test split    39 no
Preproces…
##  9 yes           0.984     0.0163 train/test split    48 yes
Preproces…
## 10 yes           0.969     0.0311 train/test split    51 yes
Preproces…
## # ℹ 1,018 more rows
```

Model 3

```r
## ROC Curve
  roc_curve(logistic_results, truth = loan_default, .pred_yes) %>%
  autoplot()
```




```
# ROC AUC
roc_auc(logistic_results, truth = loan_default, .pred_yes)
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.986
```

```
# Confusion Matrix
conf_mat(logistic_results, truth = loan_default, .pred_class)
```

```
##            Truth
## Prediction yes  no
##        yes 352  21
##        no   31 624
```

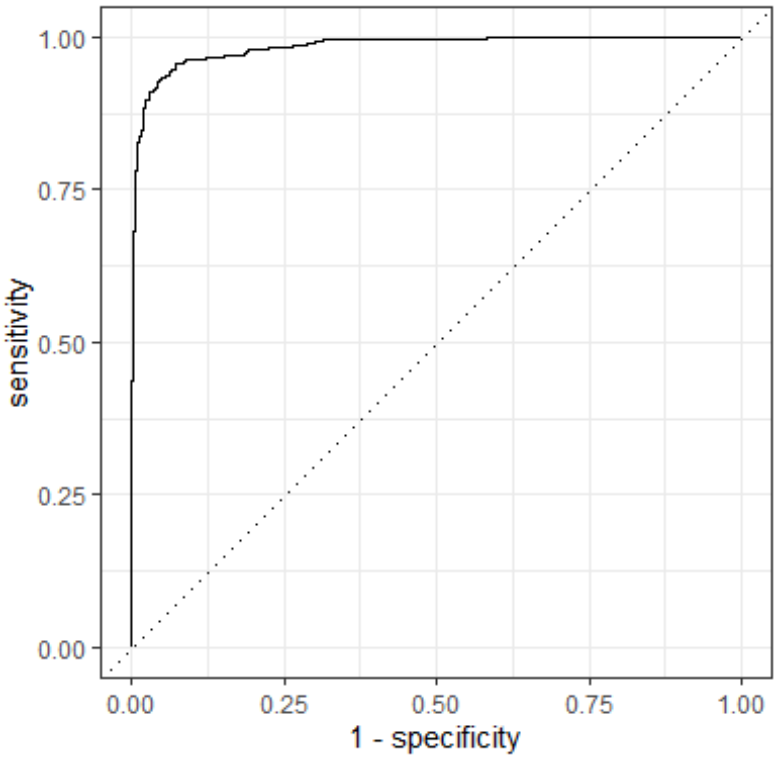
```
lda_model <- discrim_regularized() %>%
             set_engine("klaR") %>%
             set_mode("classification")

lda_model
```

```
## Regularized Discriminant Model Specification (classification)
##
## Computational engine: klaR
```

```
lda_wf <- workflow() %>%
  add_model(lda_model) %>%
```

```
  add_recipe(loan_recipe)

lda_wf

## == Workflow
==========================================================================
## Preprocessor: Recipe
## Model: discrim_regularized()
##
## -- Preprocessor
--------------------------------------------------------------------------
## 3 Recipe Steps
##
## • step_YeoJohnson()
## • step_normalize()
## • step_dummy()
##
## -- Model
---------------------------------------------------------------------------
## Regularized Discriminant Model Specification (classification)
##
## Computational engine: klaR

lda_fit <- lda_wf %>%
                fit(data = loan_training)

lda_fit

## == Workflow [trained]
==========================================================================
## Preprocessor: Recipe
## Model: discrim_regularized()
##
## -- Preprocessor
--------------------------------------------------------------------------
## 3 Recipe Steps
##
## • step_YeoJohnson()
## • step_normalize()
## • step_dummy()
##
## -- Model
---------------------------------------------------------------------------
## Call:
## rda(formula = ..y ~ ., data = data)
##
## Regularization parameters:
##        gamma       lambda
## 1.039327e-22 9.998389e-01
##
```

```
## Prior probabilities of groups:
##       yes        no
## 0.3721609 0.6278391
##
## Misclassification rate:
##        apparent: 5.516 %
## cross-validated: 5.679 %

lda_final <- lda_wf %>%
  last_fit(split = loan_split)

lda_results <- lda_final %>%
  collect_predictions()

lda_results

## # A tibble: 1,028 × 7
##    .pred_class .pred_yes .pred_no id                .row loan_default
.config
##    <fct>           <dbl>    <dbl> <chr>            <int> <fct>
<chr>
##  1 no            0.0522    0.948  train/test split     8 no
Preproces…
##  2 yes           0.693     0.307  train/test split    15 yes
Preproces…
##  3 yes           0.690     0.310  train/test split    21 yes
Preproces…
##  4 yes           0.684     0.316  train/test split    25 yes
Preproces…
##  5 no            0.0712    0.929  train/test split    28 no
Preproces…
##  6 yes           0.941     0.0589 train/test split    32 yes
Preproces…
##  7 no            0.00984   0.990  train/test split    38 no
Preproces…
##  8 no            0.0283    0.972  train/test split    39 no
Preproces…
##  9 yes           0.947     0.0526 train/test split    48 yes
Preproces…
## 10 yes           0.861     0.139  train/test split    51 yes
Preproces…
## # ℹ 1,018 more rows

## ROC Curve
roc_curve(lda_results, truth = loan_default, .pred_yes) %>%
  autoplot()
```




```
# ROC AUC
roc_auc(lda_results, truth = loan_default, .pred_yes)
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.984
```

```
# Confusion Matrix
conf_mat(lda_results, truth = loan_default, .pred_class)
```

```
##           Truth
## Prediction yes  no
##        yes 343  15
##        no   40 630
```

```
knn_model <- nearest_neighbor(neighbors = tune()) %>%
             set_engine("kknn") %>%
             set_mode("classification")
knn_model
```

```
## K-Nearest Neighbor Model Specification (classification)
##
## Main Arguments:
##   neighbors = tune()
##
## Computational engine: kknn
```

```r
knn_wf <- workflow() %>%
          add_model(knn_model) %>%
          add_recipe(loan_recipe)

knn_wf
```

```
## == Workflow
═══════════════════════════════════════════════════════════
## Preprocessor: Recipe
## Model: nearest_neighbor()
##
## ── Preprocessor
──────────────────────────────────────────────────────────
## 3 Recipe Steps
##
## • step_YeoJohnson()
## • step_normalize()
## • step_dummy()
##
## ── Model
──────────────────────────────────────────────────────────
## K-Nearest Neighbor Model Specification (classification)
##
## Main Arguments:
##   neighbors = tune()
##
## Computational engine: kknn
```

```r
# Tuning Grid for KNN with random grid and cross-validation

# Load required packages
library(tune)

# Define tuning grid for k-NN (neighbors)
knn_params <- parameters(knn_model)
```

```
## Warning: `parameters.model_spec()` was deprecated in tune 0.1.6.9003.
## ℹ Please use `hardhat::extract_parameter_set_dials()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```r
# Create a random grid of 10 values
set.seed(123)
knn_grid <- grid_random(knn_params, size = 10)

# Tune the model using cross-validation and random search
knn_tune_results <- tune_grid(
  knn_wf,
  resamples = loan_folds,
```

```
  grid = knn_grid,
  metrics = metric_set(roc_auc)
)
```

```
## → A | warning: variable '..y' is absent, its contrast will be ignored
```

```
## There were issues with some computations   A: x1There were issues with
some computations   A: x2There were issues with some computations   A:
x3There were issues with some computations   A: x4There were issues with some
computations   A: x5There were issues with some computations   A: x5
```

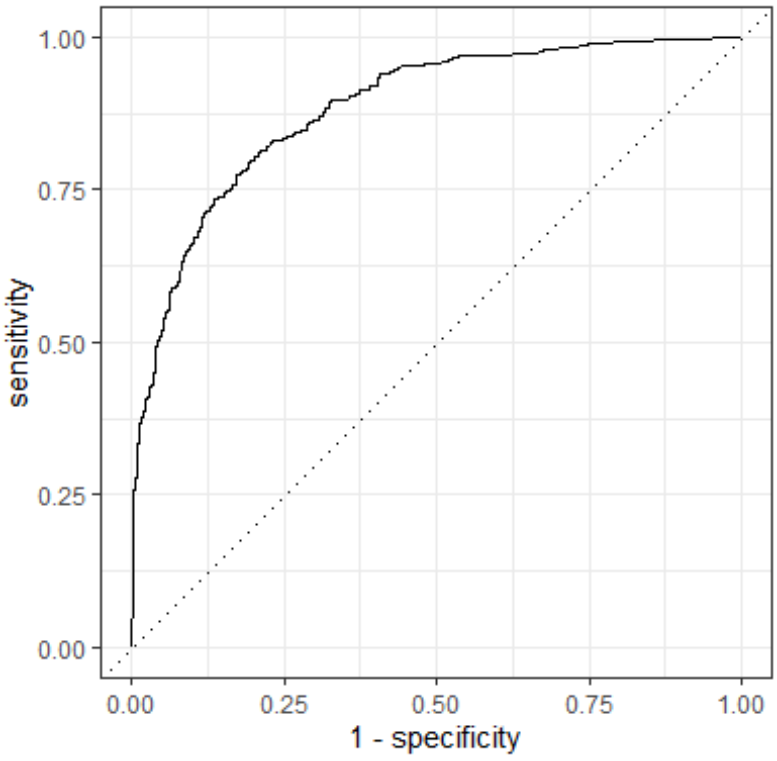
```
# Show best results
show_best(knn_tune_results, metric = "roc_auc")
```

```
## # A tibble: 5 × 7
##   neighbors .metric .estimator  mean     n std_err .config
##       <int> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
## 1        15 roc_auc binary     0.868     5 0.00384 Preprocessor1_Model8
## 2        14 roc_auc binary     0.866     5 0.00385 Preprocessor1_Model7
## 3        11 roc_auc binary     0.857     5 0.00400 Preprocessor1_Model6
## 4        10 roc_auc binary     0.855     5 0.00406 Preprocessor1_Model5
## 5         6 roc_auc binary     0.836     5 0.00311 Preprocessor1_Model4
```

```
# Select best model
best_knn <- select_best(knn_tune_results, metric = "roc_auc")

# Finalize the workflow with best hyperparameters
final_knn_wf <- finalize_workflow(knn_wf, best_knn)

# Fit on the training data and evaluate on the test split
knn_final_fit <- final_knn_wf %>%
  last_fit(split = loan_split)
```

```
## → A | warning: variable '..y' is absent, its contrast will be ignored
## There were issues with some computations   A: x1There were issues with
some computations   A: x1
```

```
# Collect and inspect predictions
knn_results <- knn_final_fit %>% collect_predictions()

# ROC Curve
roc_curve(knn_results, truth = loan_default, .pred_yes) %>% autoplot()
```




```
# ROC AUC
roc_auc(knn_results, truth = loan_default, .pred_yes)
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.883
```

```
# Confusion Matrix
conf_mat(knn_results, truth = loan_default, .pred_class)
```

```
##            Truth
## Prediction yes  no
##        yes 245  56
##        no  138 589
```

— End of the Project —