Final Project

**Your Name**: YEEUN PARK **Your G Number**: G01176416

# Suppress dplyr summarise grouping warning messages  
options(dplyr.summarise.inform = FALSE)  
  
## Add R libraries here  
library(tidyverse)  
library(tidymodels)  
library(vip)  
library(parsnip)  
library(recipes)  
library(rpart.plot)  
library(ranger)  
library(discrim)  
# Load data  
loans\_df <- read\_rds("C:/Users/ssiba/Downloads/loan\_data.rds")  
  
loans\_df

## # A tibble: 4,110 × 16  
## loan\_…¹ loan\_…² insta…³ inter…⁴ loan\_…⁵ appli…⁶ term homeo…⁷ annua…⁸ curre…⁹  
## <fct> <int> <dbl> <dbl> <fct> <fct> <fct> <fct> <dbl> <dbl>  
## 1 yes 35000 927. 17.2 small\_… indivi… five… rent 104660 2  
## 2 yes 10000 260. 11.5 small\_… indivi… five… mortga… 57000 10  
## 3 no 28800 942. 8.97 debt\_c… indivi… thre… rent 160000 10  
## 4 yes 4475 165. 10 medical indivi… thre… rent 37000 1  
## 5 no 3600 111. 9.72 medical indivi… thre… mortga… 72000 4  
## 6 yes 12800 389. 20 medical indivi… five… rent 73000 10  
## 7 yes 35000 927. 18.2 debt\_c… indivi… five… mortga… 167000 0  
## 8 no 26000 619. 12.0 debt\_c… indivi… five… mortga… 125000 5  
## 9 no 5500 176. 7.97 debt\_c… indivi… thre… rent 70000 4  
## 10 no 40000 952. 11.0 home\_i… indivi… five… mortga… 70000 3  
## # … with 4,100 more rows, 6 more variables: debt\_to\_income <dbl>,  
## # total\_credit\_lines <int>, years\_credit\_history <dbl>,  
## # missed\_payment\_2\_yr <fct>, history\_bankruptcy <fct>,  
## # history\_tax\_liens <fct>, and abbreviated variable names ¹​loan\_default,  
## # ²​loan\_amount, ³​installment, ⁴​interest\_rate, ⁵​loan\_purpose,  
## # ⁶​application\_type, ⁷​homeownership, ⁸​annual\_income, ⁹​current\_job\_years

# Summary of Results [50 Points]

Write a summary of your overall findings and recommendations to the executives at the bank. Think of this section as your closing remarks of a presentation, where you summarize your key findings, model performance, and make recommendations to improve loan processes at the bank.

Your executive summary must be written in a business tone, with minimal grammatical errors, and should include the following sections:

## 1. An introduction where you explain the business problem and goals of your data analysis

The National Bank provides loans to bank customers. Banks should prevent financial losses by distinguishing between customers who default on loans. The more customers default on their loans, the more financial losses and risks the bank may suffer because it cannot get money back from the customers. They can provide loans based on risk by distinguishing customers who have a probability of defaulting on loans. It can increase the amount of loans to customers who are less likely to default on loans. Also, they do not provide loans to customers who are more likely to default on loans. By predicting in advance whether or not the customer will default on the loan, it is possible to make a decision accordingly and provide the service. This is a precautionary measure to prevent a bank’s financial losses and an effective way to improve its financial benefits and efficiency.

The bank’s goal is to better identify customers who are at risk of loan default to minimize financial losses. In order to distinguish them, information obtained from customers can be analyzed through several assumptions. Here are six assumptions that set up customers with information:

1. Are there differences in loan default rates by annual income and home ownership?
2. Are there differences in loan default rates by years of credit history?
3. Are there differences in loan default rates by monthly payment amount?
4. Are there differences in loan default rates by debt to income ratio at application time and history of missed payments in the last 2 years?
5. Are there differences in loan default rates by interest rate and loan application type?
6. Are there differences in loan default rates by the history of bankruptcy?

Through these assumptions, it is possible to analyze the correlation of information and determine the main factors that lead to loan defaults. Various assumptions and analyses can identify the main factors, establish conditions for customers to be provided loans, and specify the amount of loans accordingly.

## 2. Highlights and key findings from your Exploratory Data Analysis section

Exploratory data analysis helps you better understand information, causes, and solutions to solve a company’s problems or to achieve its goals. They provide a direction to solve by helping to identify the main cause. This helps you identify and obtain more accurate information. While searching for data, we examine the relationship between information and compare and analyze what relationship is important.

1. The relationship between home ownership, annual income, and default on loans seems a little important. Borrowers whose homes are mortgage or rent are affected by their annual income. In other words, those with low annual income often suffer from loan defaults. However, homeowners are not affected by their annual income. In other words, even if the annual income is low, they often do not suffer from loan defaults.
2. The relationship between credit history and loan defaults seems important. The annual minimum credit history did not affect significantly, but the average and maximum credit history did. Lenders with a low credit history are likely to default on loans.
3. The relationship between the amount of monthly installments and loan defaults seems important. The larger the amount of monthly installments, the higher the probability that the borrower defaults on the loan.
4. The relationship between the history of missed payments and loan default over the past two years is not important, but the relationship between debt to income ratio and loan default is important. There has been no difference between borrowers with and without a history of unpaid bills in the past two years, and there have been more default on loans by borrowers without a history of unpaid bills. On the other hand, the ratio of debt to income had a very significant impact on loan defaults. The higher the debt ratio, the higher the probability of loan defaults.
5. The type of loan application was not related to loan default. But interest rates have had a major impact on loan defaults. The higher the interest rate, the higher the probability of defaulting on the loan.
6. The relationship between the history of bankruptcy and loan default is not important. There is no correlation between them. The ratio of loan default based on bankruptcy is similar to each other.

## 3. Your “best” classification model and an analysis of its performance

In order to find the most significant factors that have the greatest impact on service termination, the most significant factors were found using the Decision Tree, the LDA model, and Random Forest. We then use ROC AUC to measure the accuracy of the model. The higher the ROC AUC, the higher the accuracy of the model prediction. The ROC AUC for the Decision Tree is 96.97%. The ROC AUC for the LDA model is 98.38%. The ROC AUC for the Random Forest model is 97.55%.After confirming that the LDA model is the best model, try to find meaningful factors in the model. According to the vip feature, there are ten factors which are Interest\_rates, loan\_amount, installment, term\_five\_year, debt\_to\_income, annual\_income, current\_job\_years, total\_credit\_lines, loan\_purpose\_medical, and loan\_purpose\_credit\_card. They greatly affect borrowers’ default on loans. Among them, Interest\_rates, loan\_amount, installment, and term\_five\_year are the highest four factors of loan defaults. Interest rate is most important and incomparable highest level.

## 4. Your recommendations to the company on how to reduce loan default rates

According to Random Forest and Decision Tree, interest\_rate is the most important factor. This means that the borrower has the greatest impact on defaulting on the loan, and I think this should be considered the most by banks. Interest\_rate shows a figure that is incomparably higher than other elements. In other words, interest rates are the main factor in borrowers’ difficulty in fulfilling their loans. Banks can increase borrowers’ performance rates by analyzing and adjusting interest rates in detail. Next, the amount of the loan and the monthly installment amount are similar, followed by interest rates. Borrowers’ monthly installments and the amount of loans directly affect their debt by the amount of money they have to pay regularly and the total amount they have to pay back each month. Looking at this, interest rates, monthly installments, and the amount of loans are directly related to money. It can be seen that money has the most influence than external factors such as whether borrowers own a house or credit history. By quantifying money-related factors, banks can see that they should pay the most attention to and analyze them.

## 5. Conclusion

Banks try to prevent financial losses and boost profits by reducing borrowers’ default on loans. In order to reduce borrowers’ loan defaults, the information of borrowers was compared and analyzed. Exploratory data analysis techniques help clarify what the main cause is and show what needs to be improved. Through this analysis, factors affecting loan defaults and factors that do not affect were found. In addition, through in-depth analysis, the factors that have the greatest influence were identified with a histogram and found commonalities.

First, it can be seen that interest rates, loan volume, and monthly installment payments are the main factors that have the greatest impact. Interest rates are the factor that most directly affects the bank’s profits, but they are the biggest factor for borrowers to default on loans. Banks should analyze the interest rate to what extent the interest rate will be the least default on loans and the highest return rate. Also, interest rates, amount of loans, and monthly installments are all directly related to money. This shows that money-related factors have the greatest influence on loan defaults than external factors.

In order to reduce the number of loan defaulters, banks can provide loans by identifying borrowers who are more likely to default. The borrower can be identified by considering the interest rate, monthly installments, and amount of loans that are directly related to money. Providing loans considering these three factors and some external factors, such as credit history, could reduce the number of defaulters.

## 6. Appendix/Appendices

# Data Analysis [30 Points]

In this section, you must think of at least 6 relevant questions that explore the relationship between loan\_default and the other variables in the loan\_df data set. The goal of your analysis should be discovering which variables drive the differences between customers who do and do not default on their loans.

You must answer each question and provide supporting data summaries with either a summary data frame (using dplyr/tidyr) or a plot (using ggplot) or both.

In total, you must have a minimum of 3 plots (created with ggplot) and 3 summary data frames (created with dplyr) for the exploratory data analysis section. Among the plots you produce, you must have at least 3 different types (ex. box plot, bar chart, histogram, scatter plot, etc…)

See the example question below.

**Note**: To add an R code chunk to any section of your project, you can use the keyboard shortcut Ctrl + Alt + i or the insert button at the top of your R project template notebook file.

## Sample Question

**Are there differences in loan default rates by loan purpose?**

**Answer**: Yes, the data indicates that credit card and medical loans have significantly larger default rates than any other type of loan. In fact, both of these loan types have default rates at more than 50%. This is nearly two times the average default rate for all other loan types.

### Summary Table

loans\_df %>%  
 group\_by(loan\_purpose) %>%   
 summarise(n\_customers = n(),  
 customers\_default = sum(loan\_default == 'yes'),  
 default\_percent = 100 \* mean(loan\_default == 'yes'))

## # A tibble: 5 × 4  
## loan\_purpose n\_customers customers\_default default\_percent  
## <fct> <int> <int> <dbl>  
## 1 debt\_consolidation 1218 308 25.3  
## 2 credit\_card 879 470 53.5  
## 3 medical 635 384 60.5  
## 4 small\_business 853 221 25.9  
## 5 home\_improvement 525 147 28

loans\_df

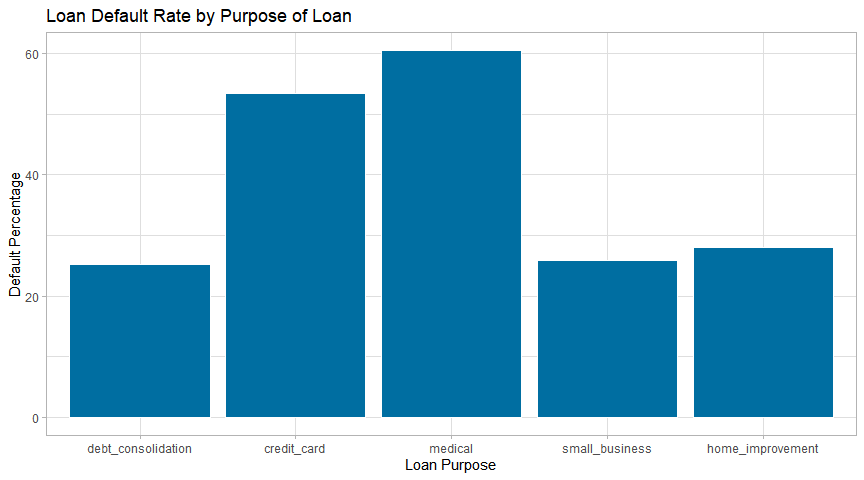
## # A tibble: 4,110 × 16  
## loan\_…¹ loan\_…² insta…³ inter…⁴ loan\_…⁵ appli…⁶ term homeo…⁷ annua…⁸ curre…⁹  
## <fct> <int> <dbl> <dbl> <fct> <fct> <fct> <fct> <dbl> <dbl>  
## 1 yes 35000 927. 17.2 small\_… indivi… five… rent 104660 2  
## 2 yes 10000 260. 11.5 small\_… indivi… five… mortga… 57000 10  
## 3 no 28800 942. 8.97 debt\_c… indivi… thre… rent 160000 10  
## 4 yes 4475 165. 10 medical indivi… thre… rent 37000 1  
## 5 no 3600 111. 9.72 medical indivi… thre… mortga… 72000 4  
## 6 yes 12800 389. 20 medical indivi… five… rent 73000 10  
## 7 yes 35000 927. 18.2 debt\_c… indivi… five… mortga… 167000 0  
## 8 no 26000 619. 12.0 debt\_c… indivi… five… mortga… 125000 5  
## 9 no 5500 176. 7.97 debt\_c… indivi… thre… rent 70000 4  
## 10 no 40000 952. 11.0 home\_i… indivi… five… mortga… 70000 3  
## # … with 4,100 more rows, 6 more variables: debt\_to\_income <dbl>,  
## # total\_credit\_lines <int>, years\_credit\_history <dbl>,  
## # missed\_payment\_2\_yr <fct>, history\_bankruptcy <fct>,  
## # history\_tax\_liens <fct>, and abbreviated variable names ¹​loan\_default,  
## # ²​loan\_amount, ³​installment, ⁴​interest\_rate, ⁵​loan\_purpose,  
## # ⁶​application\_type, ⁷​homeownership, ⁸​annual\_income, ⁹​current\_job\_years

### Data Visulatization

default\_rates <- loans\_df %>%  
 group\_by(loan\_purpose) %>%   
 summarise(n\_customers = n(),  
 customers\_default = sum(loan\_default == 'yes'),  
 default\_percent = 100 \* mean(loan\_default == 'yes'))  
  
default\_rates

## # A tibble: 5 × 4  
## loan\_purpose n\_customers customers\_default default\_percent  
## <fct> <int> <int> <dbl>  
## 1 debt\_consolidation 1218 308 25.3  
## 2 credit\_card 879 470 53.5  
## 3 medical 635 384 60.5  
## 4 small\_business 853 221 25.9  
## 5 home\_improvement 525 147 28

ggplot(data = default\_rates, mapping = aes(x = loan\_purpose, y = default\_percent)) +  
 geom\_bar(stat = 'identity', fill = '#006EA1', color = 'white') +  
 labs(title = 'Loan Default Rate by Purpose of Loan',  
 x = 'Loan Purpose',  
 y = 'Default Percentage') +  
 theme\_light()



## Question 1

**Question**: Are there differences in loan default rates by annual income and home ownership?

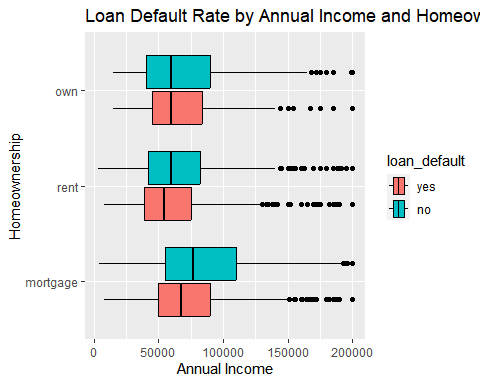
**Answer**: Yes, there are three types of home ownership which are rent, mortgage, and own. For mortgages or rent, customers who had loan defaults are lower annual incomes than those who had no loan defaults. This means that annual income affected to loan defaults. On the other hand, people who own homes were not affected by their annual income.

default\_rates <- loans\_df %>%  
 group\_by(loan\_default, homeownership) %>%   
 summarise(n\_customers = n(),  
 quantile1\_annual\_income = quantile(annual\_income, 1 / 4),  
 quantile2\_annual\_income = quantile(annual\_income, 2 / 4),  
 quantile3\_annual\_income = quantile(annual\_income, 3 / 4)) %>%  
 arrange(homeownership)  
  
  
default\_rates

## # A tibble: 6 × 6  
## # Groups: loan\_default [2]  
## loan\_default homeownership n\_customers quantile1\_annual\_income quant…¹ quant…²  
## <fct> <fct> <int> <dbl> <dbl> <dbl>  
## 1 yes mortgage 628 49627. 67250 90000  
## 2 no mortgage 1309 55000 77000 110000  
## 3 yes rent 713 39000 54200 75000  
## 4 no rent 953 42000 60000 82000  
## 5 yes own 189 45000 60000 84000  
## 6 no own 318 40000 60000 90000  
## # … with abbreviated variable names ¹​quantile2\_annual\_income,  
## # ²​quantile3\_annual\_income

ggplot(data = loans\_df, mapping = aes(x = annual\_income, fill = loan\_default)) +  
 geom\_boxplot(aes(y = homeownership), color = "black", bins = 30) +  
 labs(title = 'Loan Default Rate by Annual Income and Homeownership',  
 x = 'Annual Income',  
 y = 'Homeownership')

## Warning in geom\_boxplot(aes(y = homeownership), color = "black", bins = 30):  
## Ignoring unknown parameters: `bins`



## Question 2

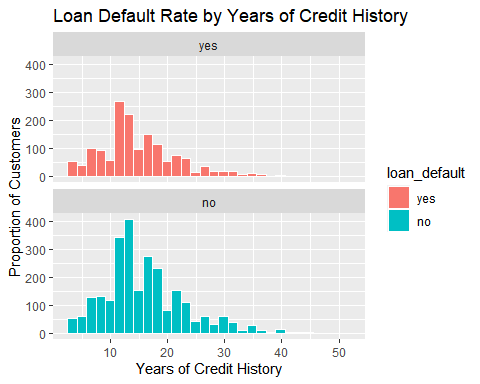
**Question**: Are there differences in loan default rates by years of credit history?

**Answer**: Yes, credit history has affected the default rate of loans. The annual minimum credit history was the same for borrowers who did not default on loans and borrowers who defaulted on loans. However, borrowers who did not default on loans were higher than borrowers who defaulted on loans for the annual maximum credit history and the annual average credit history. This means that it is a higher probability for borrowers with high credit history to not default on loans.

default\_rates <- loans\_df %>%  
 group\_by(loan\_default) %>%   
 summarise(n\_customers = n(),  
 min\_years\_credit\_history = min(years\_credit\_history),  
 max\_years\_credit\_history = max(years\_credit\_history),  
 avg\_years\_credit\_history = mean(years\_credit\_history),  
 sd\_years\_credit\_history = sd(years\_credit\_history))  
   
  
  
default\_rates

## # A tibble: 2 × 6  
## loan\_default n\_customers min\_years\_credit\_history max\_years\_…¹ avg\_y…² sd\_ye…³  
## <fct> <int> <dbl> <dbl> <dbl> <dbl>  
## 1 yes 1530 3 46 15.0 6.86  
## 2 no 2580 3 51 16.2 7.38  
## # … with abbreviated variable names ¹​max\_years\_credit\_history,  
## # ²​avg\_years\_credit\_history, ³​sd\_years\_credit\_history

ggplot(data = loans\_df, aes(x = years\_credit\_history, fill = loan\_default))+  
 geom\_histogram(color = "white", bins = 30)+  
 facet\_wrap(~ loan\_default, nrow = 2) +  
 labs(title = "Loan Default Rate by Years of Credit History",  
 x = "Years of Credit History", y = "Proportion of Customers")



## Question 3

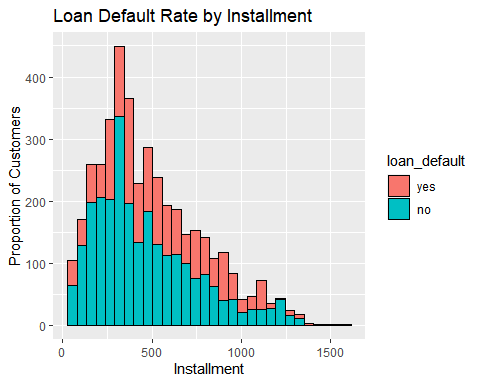
**Question**: Are there differences in loan default rates by monthly payment amount?

**Answer**: Yes, the monthly installments of default borrowers were higher in all quarters. It shows that the higher the monthly installment, the more difficult it is for the borrower to fulfill the debt. It means that borrowers with higher monthly installments are more likely to default on their debts.

default\_rates <- loans\_df %>%  
 group\_by(loan\_default) %>%   
 summarise(n\_customers = n(),  
 quantile1\_installment = quantile(installment, 1 / 4),  
 quantile2\_installment = quantile(installment, 2 / 4),  
 quantile3\_installment = quantile(installment, 3 / 4))  
  
  
default\_rates

## # A tibble: 2 × 5  
## loan\_default n\_customers quantile1\_installment quantile2\_installment quantil…¹  
## <fct> <int> <dbl> <dbl> <dbl>  
## 1 yes 1530 309. 492. 747.  
## 2 no 2580 251. 387. 627.  
## # … with abbreviated variable name ¹​quantile3\_installment

ggplot(data = loans\_df, mapping = aes(x = installment, fill = loan\_default)) +  
 geom\_histogram(color = "black", bins = 30) +  
 labs(title = 'Loan Default Rate by Installment',  
 x = 'Installment',  
 y = 'Proportion of Customers')



## Question 4

**Question**: Are there differences in loan default rates by debt to income ratio at application time and history of missed payments in the last 2 years?

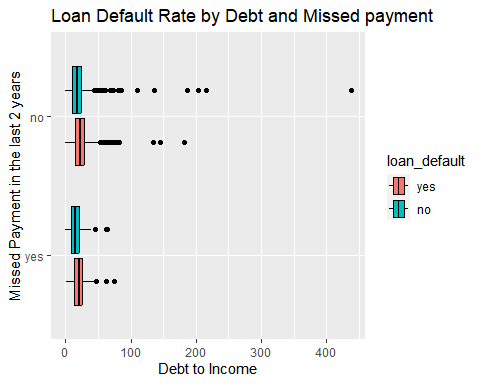
**Answer**: First, the presence or absence of an experience of missing payments over the past two years is considered irrelevant to loan default. This is because the number of default borrowers who experienced missing payments in the last two years was less than those who inexperienced. In other words, it means that borrowers who have not missed payments in the past two years have been more in default. It shows that the history of the past two years has nothing to do with loan defaults. Next, the debt ratio at the time of application affected the loan default. In all quarters, the higher the debt ratio at the time of application, the more the loan defaulted. It shows that the debt ratio has a significant impact on loan defaults.

default\_rates <- loans\_df %>%  
 group\_by(loan\_default, missed\_payment\_2\_yr) %>%   
 summarise(n\_customers = n(),  
 quantile1\_debt\_to\_income = quantile(debt\_to\_income, 1 / 4),  
 quantile2\_debt\_to\_income = quantile(debt\_to\_income, 2 / 4),  
 quantile3\_debt\_to\_income = quantile(debt\_to\_income, 3 / 4)) %>%  
 arrange(missed\_payment\_2\_yr)  
  
  
default\_rates

## # A tibble: 4 × 6  
## # Groups: loan\_default [2]  
## loan\_default missed\_payment\_2\_yr n\_customers quantile1\_debt\_…¹ quant…² quant…³  
## <fct> <fct> <int> <dbl> <dbl> <dbl>  
## 1 yes yes 212 12.9 20.3 26.3  
## 2 no yes 258 9.36 15.0 21.5  
## 3 yes no 1318 14.4 21.8 29.4  
## 4 no no 2322 10.9 17.3 24.4  
## # … with abbreviated variable names ¹​quantile1\_debt\_to\_income,  
## # ²​quantile2\_debt\_to\_income, ³​quantile3\_debt\_to\_income

ggplot(data = loans\_df, mapping = aes(x = debt\_to\_income, fill = loan\_default)) +  
 geom\_boxplot(aes(y = missed\_payment\_2\_yr), color = "black", bins = 30) +  
 labs(title = 'Loan Default Rate by Debt and Missed payment',  
 x = 'Debt to Income',  
 y = 'Missed Payment in the last 2 years')

## Warning in geom\_boxplot(aes(y = missed\_payment\_2\_yr), color = "black", bins =  
## 30): Ignoring unknown parameters: `bins`



## Question 5

**Question**: Are there differences in loan default rates by interest rate and loan application type?

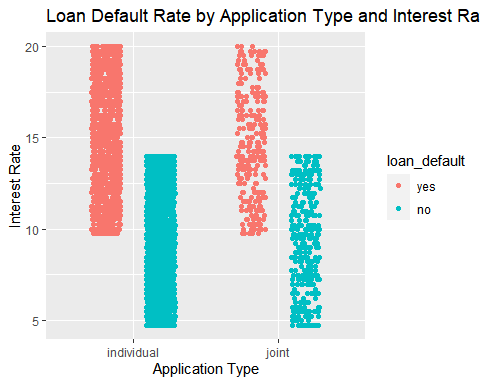
**Answer**: First, whether or not the borrower defaults on the debt, the number of borrowers with individual loan application types was much higher than the number of joint borrowers. In addition, the higher the interest rate in all quarters, whether individual or joint borrowers, the higher the loan default. It shows that the higher the interest rate, the higher the probability of defaulting on the loan.

default\_rates <- loans\_df %>%  
 group\_by(loan\_default, application\_type) %>%   
 summarise(n\_customers = n(),  
 min\_interest\_rate = min(interest\_rate),  
 max\_interest\_rate = max(interest\_rate),  
 avg\_interest\_rate = mean(interest\_rate),  
 sd\_interest\_rate = sd(interest\_rate))%>%  
 arrange(application\_type)  
   
  
  
default\_rates

## # A tibble: 4 × 7  
## # Groups: loan\_default [2]  
## loan\_default application\_type n\_customers min\_intere…¹ max\_i…² avg\_i…³ sd\_in…⁴  
## <fct> <fct> <int> <dbl> <dbl> <dbl> <dbl>  
## 1 yes individual 1253 9.75 20 14.9 3.01  
## 2 no individual 2241 4.72 14.0 9.32 2.75  
## 3 yes joint 277 9.75 20 14.7 2.93  
## 4 no joint 339 4.72 14.0 9.20 2.79  
## # … with abbreviated variable names ¹​min\_interest\_rate, ²​max\_interest\_rate,  
## # ³​avg\_interest\_rate, ⁴​sd\_interest\_rate

ggplot(data = loans\_df, aes(x= application\_type, y = interest\_rate, color = loan\_default))+  
 geom\_point(position=position\_jitterdodge(), bins = 10) +  
 labs(title = "Loan Default Rate by Application Type and Interest Rate",  
 x = "Application Type", y = "Interest Rate")

## Warning in geom\_point(position = position\_jitterdodge(), bins = 10): Ignoring  
## unknown parameters: `bins`



## Question 6

**Question**: Are there differences in loan default rates by the history of bankruptcy?

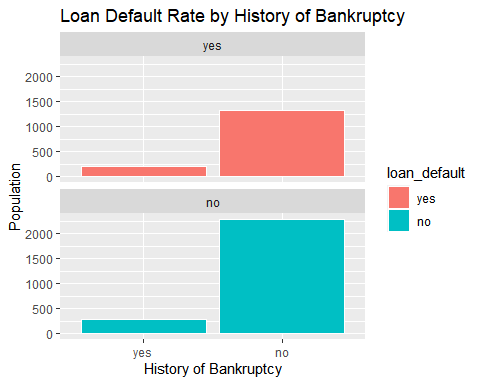
**Answer**: No, there are no large differences in loan default rates between the presence or absence of bankruptcy history. Either the borrower has a default on loan or not, the ratio of loan defaults based on bankruptcy is same.

default\_rates <- loans\_df %>%  
 group\_by(loan\_default, history\_bankruptcy) %>%   
 summarise(total = n())  
  
  
default\_rates

## # A tibble: 4 × 3  
## # Groups: loan\_default [2]  
## loan\_default history\_bankruptcy total  
## <fct> <fct> <int>  
## 1 yes yes 203  
## 2 yes no 1327  
## 3 no yes 283  
## 4 no no 2297

ggplot(data = default\_rates, aes(x= history\_bankruptcy, fill = loan\_default))+  
 geom\_histogram(stat = "identity", aes(y = total), color = "white") +  
 facet\_wrap(~ loan\_default, nrow = 2) +  
 labs(title = "Loan Default Rate by History of Bankruptcy ",  
 x = "History of Bankruptcy", y = "Population")

## Warning in geom\_histogram(stat = "identity", aes(y = total), color = "white"):  
## Ignoring unknown parameters: `binwidth`, `bins`, and `pad`



# Predictive Modeling [70 Points]

In this section of the project, you will fit **three classification algorithms** to predict the response variable,loan\_default. You should use all of the other variables in the loans\_df data as predictor variables for each model.

You must follow the machine learning steps below.

The data splitting and feature engineering steps should only be done once so that your models are using the same data and feature engineering steps for training.

* Split the loans\_df data into a training and test set (remember to set your seed)
* Specify a feature engineering pipeline with the recipes package
  + You can include steps such as skewness transformation, dummy variable encoding or any other steps you find appropriate
* Specify a parsnip model object
  + You may choose from the following classification algorithms:
    - Logistic Regression
    - LDA
    - QDA
    - KNN
    - Decision Tree
    - Random Forest
* Package your recipe and model into a workflow
* Fit your workflow to the training data
  + If your model has hyperparameters:
    - Split the training data into 5 folds for 5-fold cross validation using vfold\_cv (remember to set your seed)
    - Perform hyperparamter tuning with a random grid search using the grid\_random() function
    - Hyperparameter tuning can take a significant amount of computing time. Be careful not to set the size argument of grid\_random() too large. I recommend size = 10 or smaller.
    - Select the best model with select\_best() and finalize your workflow
* Evaluate model performance on the test set by plotting an ROC curve using autoplot() and calculating the area under the ROC curve on your test data

## Model 1 Decision Tree

loans\_data <- read\_rds("C:/Users/ssiba/Downloads/loan\_data.rds")  
  
loans\_split <- initial\_split(loans\_data, prop = 0.75,   
 strata = loan\_default)  
  
loans\_training <- loans\_split %>%   
 training()  
  
loans\_test <- loans\_split %>%   
 testing()  
  
loans\_folds <- vfold\_cv(loans\_training , v = 5)  
  
loans\_recipe <- recipe(loan\_default ~ ., data = loans\_training) %>%   
 step\_YeoJohnson(all\_numeric(), -all\_outcomes()) %>%   
 step\_normalize(all\_numeric(), -all\_outcomes()) %>%   
 step\_dummy(all\_nominal(), -all\_outcomes())  
  
loans\_recipe %>%   
 prep() %>%   
 bake(new\_data = loans\_training)

## # A tibble: 3,082 × 20  
## loan\_amount install…¹ inter…² annua…³ curre…⁴ debt\_…⁵ total…⁶ years…⁷ loan\_…⁸  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <fct>   
## 1 1.16 1.40 -0.559 1.88 1.09 -1.40 1.08 0.330 no   
## 2 -1.61 -1.69 -0.352 0.170 -0.400 0.406 1.08 -0.588 no   
## 3 0.960 0.592 0.229 1.32 -0.126 -1.16 0.323 0.195 no   
## 4 -1.22 -1.21 -0.847 0.115 -0.400 -0.683 -0.874 -0.972 no   
## 5 1.87 1.43 -0.0225 0.115 -0.691 -2.33 -0.874 -0.414 no   
## 6 0.0188 0.0589 0.0413 -0.834 1.09 0.763 -0.406 0.933 no   
## 7 1.63 0.983 -1.07 2.23 1.09 0.401 1.14 0.330 no   
## 8 -1.13 -1.09 0.0413 -1.35 1.09 -1.34 -0.515 -0.588 no   
## 9 -0.363 -0.835 -0.921 -1.18 -0.126 1.36 0.552 -0.0942 no   
## 10 -1.05 -1.04 -0.152 -0.450 -0.691 -0.110 1.58 0.0539 no   
## # … with 3,072 more rows, 11 more variables: 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>,  
## # missed\_payment\_2\_yr\_no <dbl>, history\_bankruptcy\_no <dbl>,  
## # history\_tax\_liens\_no <dbl>, and abbreviated variable names ¹​installment,  
## # ²​interest\_rate, ³​annual\_income, ⁴​current\_job\_years, ⁵​debt\_to\_income, …

loans\_tree\_model <- decision\_tree(cost\_complexity = tune(),  
 tree\_depth = tune(),  
 min\_n = tune()) %>%   
 set\_engine('rpart') %>%   
 set\_mode('classification')  
loans\_tree\_model

## Decision Tree Model Specification (classification)  
##   
## Main Arguments:  
## cost\_complexity = tune()  
## tree\_depth = tune()  
## min\_n = tune()  
##   
## Computational engine: rpart

loans\_tree\_workflow <- workflow() %>%   
 add\_model(loans\_tree\_model) %>%   
 add\_recipe(loans\_recipe)  
loans\_tree\_workflow

## ══ Workflow ════════════════════════════════════════════════════════════════════  
## Preprocessor: Recipe  
## Model: decision\_tree()  
##   
## ── Preprocessor ────────────────────────────────────────────────────────────────  
## 3 Recipe Steps  
##   
## • step\_YeoJohnson()  
## • step\_normalize()  
## • step\_dummy()  
##   
## ── Model ───────────────────────────────────────────────────────────────────────  
## Decision Tree Model Specification (classification)  
##   
## Main Arguments:  
## cost\_complexity = tune()  
## tree\_depth = tune()  
## min\_n = tune()  
##   
## Computational engine: rpart

loans\_tree\_grid <- grid\_regular(cost\_complexity(),  
 tree\_depth(),  
 min\_n(),  
 levels = 2)  
loans\_tree\_grid

## # A tibble: 8 × 3  
## cost\_complexity tree\_depth min\_n  
## <dbl> <int> <int>  
## 1 0.0000000001 1 2  
## 2 0.1 1 2  
## 3 0.0000000001 15 2  
## 4 0.1 15 2  
## 5 0.0000000001 1 40  
## 6 0.1 1 40  
## 7 0.0000000001 15 40  
## 8 0.1 15 40

set.seed(10)  
  
loans\_tree\_tuning <- loans\_tree\_workflow %>%   
 tune\_grid(resamples = loans\_folds,  
 grid = loans\_tree\_grid)  
loans\_tree\_tuning

## # Tuning results  
## # 5-fold cross-validation   
## # A tibble: 5 × 4  
## splits id .metrics .notes   
## <list> <chr> <list> <list>   
## 1 <split [2465/617]> Fold1 <tibble [16 × 7]> <tibble [0 × 3]>  
## 2 <split [2465/617]> Fold2 <tibble [16 × 7]> <tibble [0 × 3]>  
## 3 <split [2466/616]> Fold3 <tibble [16 × 7]> <tibble [0 × 3]>  
## 4 <split [2466/616]> Fold4 <tibble [16 × 7]> <tibble [0 × 3]>  
## 5 <split [2466/616]> Fold5 <tibble [16 × 7]> <tibble [0 × 3]>

loans\_tree\_tuning %>% show\_best('roc\_auc')

## # A tibble: 5 × 9  
## cost\_complexity tree\_depth min\_n .metric .estima…¹ mean n std\_err .config  
## <dbl> <int> <int> <chr> <chr> <dbl> <int> <dbl> <chr>   
## 1 0.0000000001 15 40 roc\_auc binary 0.958 5 0.00304 Prepro…  
## 2 0.0000000001 15 2 roc\_auc binary 0.894 5 0.00955 Prepro…  
## 3 0.0000000001 1 2 roc\_auc binary 0.801 5 0.00598 Prepro…  
## 4 0.1 1 2 roc\_auc binary 0.801 5 0.00598 Prepro…  
## 5 0.1 15 2 roc\_auc binary 0.801 5 0.00598 Prepro…  
## # … with abbreviated variable name ¹​.estimator

loans\_best\_tree <- loans\_tree\_tuning %>%   
 select\_best(metric = 'roc\_auc')  
loans\_best\_tree

## # A tibble: 1 × 4  
## cost\_complexity tree\_depth min\_n .config   
## <dbl> <int> <int> <chr>   
## 1 0.0000000001 15 40 Preprocessor1\_Model7

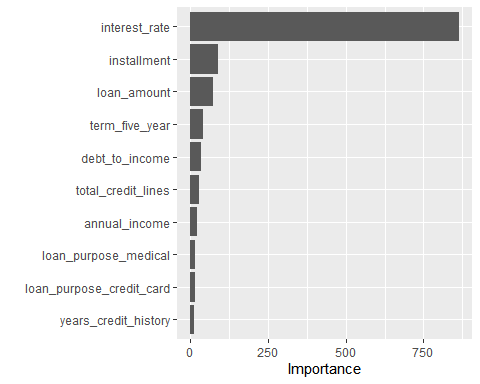
loans\_final\_tree\_workflow <- loans\_tree\_workflow %>%   
 finalize\_workflow(loans\_best\_tree)  
loans\_final\_tree\_workflow

## ══ Workflow ════════════════════════════════════════════════════════════════════  
## Preprocessor: Recipe  
## Model: decision\_tree()  
##   
## ── Preprocessor ────────────────────────────────────────────────────────────────  
## 3 Recipe Steps  
##   
## • step\_YeoJohnson()  
## • step\_normalize()  
## • step\_dummy()  
##   
## ── Model ───────────────────────────────────────────────────────────────────────  
## Decision Tree Model Specification (classification)  
##   
## Main Arguments:  
## cost\_complexity = 1e-10  
## tree\_depth = 15  
## min\_n = 40  
##   
## Computational engine: rpart

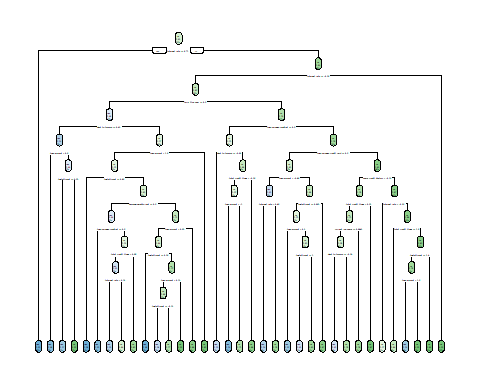
loans\_tree\_wf\_fit <- loans\_final\_tree\_workflow %>%   
 fit(data = loans\_training)  
loans\_tree\_fit <- loans\_tree\_wf\_fit %>%   
 pull\_workflow\_fit()

## Warning: `pull\_workflow\_fit()` was deprecated in workflows 0.2.3.  
## ℹ Please use `extract\_fit\_parsnip()` instead.

vip(loans\_tree\_fit)



rpart.plot(loans\_tree\_fit$fit, roundint = FALSE)



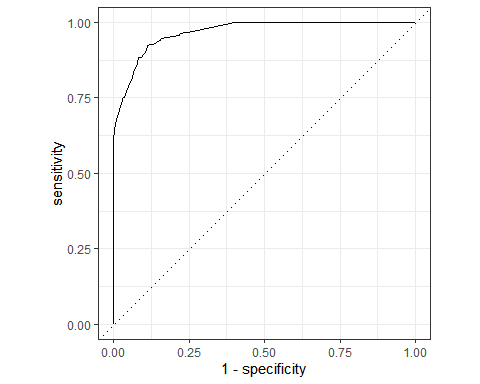
loans\_tree\_last\_fit <- loans\_final\_tree\_workflow %>%   
 last\_fit(loans\_split)  
loans\_tree\_last\_fit %>% collect\_metrics()

## # A tibble: 2 × 4  
## .metric .estimator .estimate .config   
## <chr> <chr> <dbl> <chr>   
## 1 accuracy binary 0.899 Preprocessor1\_Model1  
## 2 roc\_auc binary 0.968 Preprocessor1\_Model1

loans\_tree\_predictions <- loans\_tree\_last\_fit %>%   
 collect\_predictions()  
loans\_tree\_predictions

## # A tibble: 1,028 × 7  
## id .pred\_yes .pred\_no .row .pred\_class loan\_default .config   
## <chr> <dbl> <dbl> <int> <fct> <fct> <chr>   
## 1 train/test split 0.786 0.214 4 yes yes Preproces…  
## 2 train/test split 0.364 0.636 17 no no Preproces…  
## 3 train/test split 0 1 19 no no Preproces…  
## 4 train/test split 0 1 22 no no Preproces…  
## 5 train/test split 1 0 41 yes yes Preproces…  
## 6 train/test split 0 1 46 no no Preproces…  
## 7 train/test split 1 0 48 yes yes Preproces…  
## 8 train/test split 0.567 0.433 49 yes yes Preproces…  
## 9 train/test split 1 0 60 yes yes Preproces…  
## 10 train/test split 0.0714 0.929 70 no no Preproces…  
## # … with 1,018 more rows

loans\_tree\_predictions %>%   
 roc\_curve(truth = loan\_default, estimate = .pred\_yes) %>% autoplot()



conf\_mat(loans\_tree\_predictions, truth = loan\_default, estimate = .pred\_class)

## Truth  
## Prediction yes no  
## yes 344 65  
## no 39 580

## Model 2 Random Forest

loans\_rf\_model <- rand\_forest(mtry = tune(),  
 trees = tune(),  
 min\_n = tune()) %>%   
 set\_engine('ranger', importance = "impurity") %>% set\_mode('classification')  
loans\_rf\_model

## Random Forest Model Specification (classification)  
##   
## Main Arguments:  
## mtry = tune()  
## trees = tune()  
## min\_n = tune()  
##   
## Engine-Specific Arguments:  
## importance = impurity  
##   
## Computational engine: ranger

loans\_rf\_workflow <- workflow() %>%   
 add\_model(loans\_rf\_model) %>%   
 add\_recipe(loans\_recipe)  
loans\_rf\_workflow

## ══ Workflow ════════════════════════════════════════════════════════════════════  
## Preprocessor: Recipe  
## Model: rand\_forest()  
##   
## ── Preprocessor ────────────────────────────────────────────────────────────────  
## 3 Recipe Steps  
##   
## • step\_YeoJohnson()  
## • step\_normalize()  
## • step\_dummy()  
##   
## ── Model ───────────────────────────────────────────────────────────────────────  
## Random Forest Model Specification (classification)  
##   
## Main Arguments:  
## mtry = tune()  
## trees = tune()  
## min\_n = tune()  
##   
## Engine-Specific Arguments:  
## importance = impurity  
##   
## Computational engine: ranger

set.seed(11)  
  
loans\_rf\_grid <- grid\_random(mtry() %>% range\_set(c(4, 12)),  
 trees(),  
 min\_n(),  
 size = 10)  
loans\_rf\_grid

## # A tibble: 10 × 3  
## mtry trees min\_n  
## <int> <int> <int>  
## 1 5 1187 7  
## 2 11 93 4  
## 3 12 747 36  
## 4 4 1063 19  
## 5 8 1581 13  
## 6 9 1314 30  
## 7 8 433 35  
## 8 9 936 4  
## 9 10 1583 11  
## 10 8 1063 17

set.seed(12)  
  
loans\_rf\_tuning <- loans\_rf\_workflow %>%   
 tune\_grid(resamples = loans\_folds,  
 grid = loans\_rf\_grid)  
loans\_rf\_tuning

## # Tuning results  
## # 5-fold cross-validation   
## # A tibble: 5 × 4  
## splits id .metrics .notes   
## <list> <chr> <list> <list>   
## 1 <split [2465/617]> Fold1 <tibble [20 × 7]> <tibble [0 × 3]>  
## 2 <split [2465/617]> Fold2 <tibble [20 × 7]> <tibble [0 × 3]>  
## 3 <split [2466/616]> Fold3 <tibble [20 × 7]> <tibble [0 × 3]>  
## 4 <split [2466/616]> Fold4 <tibble [20 × 7]> <tibble [0 × 3]>  
## 5 <split [2466/616]> Fold5 <tibble [20 × 7]> <tibble [0 × 3]>

loans\_rf\_tuning %>% show\_best('roc\_auc')

## # A tibble: 5 × 9  
## mtry trees min\_n .metric .estimator mean n std\_err .config   
## <int> <int> <int> <chr> <chr> <dbl> <int> <dbl> <chr>   
## 1 10 1583 11 roc\_auc binary 0.971 5 0.00241 Preprocessor1\_Model09  
## 2 9 936 4 roc\_auc binary 0.971 5 0.00252 Preprocessor1\_Model08  
## 3 11 93 4 roc\_auc binary 0.970 5 0.00224 Preprocessor1\_Model02  
## 4 8 1581 13 roc\_auc binary 0.969 5 0.00245 Preprocessor1\_Model05  
## 5 8 1063 17 roc\_auc binary 0.969 5 0.00240 Preprocessor1\_Model10

loans\_best\_rf <- loans\_rf\_tuning %>%   
 select\_best(metric = 'roc\_auc')  
loans\_best\_rf

## # A tibble: 1 × 4  
## mtry trees min\_n .config   
## <int> <int> <int> <chr>   
## 1 10 1583 11 Preprocessor1\_Model09

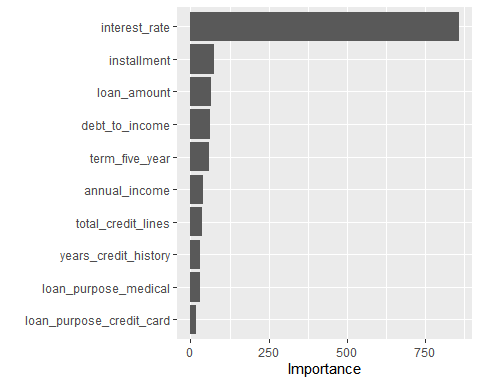
loans\_final\_rf\_workflow <- loans\_rf\_workflow %>%   
 finalize\_workflow(loans\_best\_rf)  
loans\_final\_rf\_workflow

## ══ Workflow ════════════════════════════════════════════════════════════════════  
## Preprocessor: Recipe  
## Model: rand\_forest()  
##   
## ── Preprocessor ────────────────────────────────────────────────────────────────  
## 3 Recipe Steps  
##   
## • step\_YeoJohnson()  
## • step\_normalize()  
## • step\_dummy()  
##   
## ── Model ───────────────────────────────────────────────────────────────────────  
## Random Forest Model Specification (classification)  
##   
## Main Arguments:  
## mtry = 10  
## trees = 1583  
## min\_n = 11  
##   
## Engine-Specific Arguments:  
## importance = impurity  
##   
## Computational engine: ranger

loans\_rf\_wf\_fit <- loans\_final\_rf\_workflow %>%   
 fit(data = loans\_training)  
  
loans\_rf\_fit <- loans\_rf\_wf\_fit %>%   
 pull\_workflow\_fit()  
loans\_rf\_fit

## parsnip model object  
##   
## Ranger result  
##   
## Call:  
## ranger::ranger(x = maybe\_data\_frame(x), y = y, mtry = min\_cols(~10L, x), num.trees = ~1583L, min.node.size = min\_rows(~11L, x), importance = ~"impurity", num.threads = 1, verbose = FALSE, seed = sample.int(10^5, 1), probability = TRUE)   
##   
## Type: Probability estimation   
## Number of trees: 1583   
## Sample size: 3082   
## Number of independent variables: 19   
## Mtry: 10   
## Target node size: 11   
## Variable importance mode: impurity   
## Splitrule: gini   
## OOB prediction error (Brier s.): 0.06438526

vip(loans\_rf\_fit)



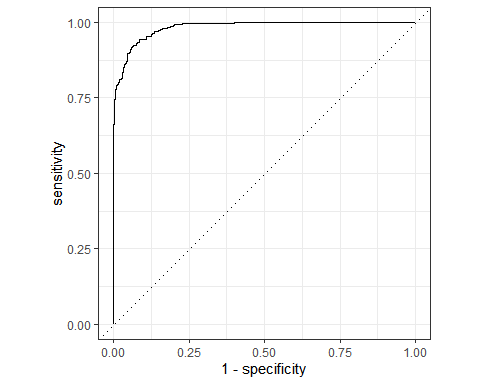
loans\_rf\_last\_fit <- loans\_final\_rf\_workflow %>%   
 last\_fit(loans\_split)  
  
loans\_rf\_last\_fit %>% collect\_metrics()

## # A tibble: 2 × 4  
## .metric .estimator .estimate .config   
## <chr> <chr> <dbl> <chr>   
## 1 accuracy binary 0.927 Preprocessor1\_Model1  
## 2 roc\_auc binary 0.983 Preprocessor1\_Model1

loans\_rf\_predictions <- loans\_rf\_last\_fit %>% collect\_predictions()  
  
loans\_rf\_predictions

## # A tibble: 1,028 × 7  
## id .pred\_yes .pred\_no .row .pred\_class loan\_default .config   
## <chr> <dbl> <dbl> <int> <fct> <fct> <chr>   
## 1 train/test split 0.532 0.468 4 yes yes Preproces…  
## 2 train/test split 0.385 0.615 17 no no Preproces…  
## 3 train/test split 0 1 19 no no Preproces…  
## 4 train/test split 0.0194 0.981 22 no no Preproces…  
## 5 train/test split 0.999 0.00126 41 yes yes Preproces…  
## 6 train/test split 0.00259 0.997 46 no no Preproces…  
## 7 train/test split 0.997 0.00328 48 yes yes Preproces…  
## 8 train/test split 0.416 0.584 49 no yes Preproces…  
## 9 train/test split 0.997 0.00277 60 yes yes Preproces…  
## 10 train/test split 0.101 0.899 70 no no Preproces…  
## # … with 1,018 more rows

loans\_rf\_predictions %>%   
 roc\_curve(truth = loan\_default, estimate = .pred\_yes) %>%   
 autoplot()



conf\_mat(loans\_rf\_predictions, truth = loan\_default, estimate = .pred\_class)

## Truth  
## Prediction yes no  
## yes 338 30  
## no 45 615

## Model 3 LDA

loans\_lda\_model <- discrim\_regularized(frac\_common\_cov = 1) %>%   
 set\_engine('klaR') %>%   
 set\_mode('classification')  
  
loans\_lda\_model

## Regularized Discriminant Model Specification (classification)  
##   
## Main Arguments:  
## frac\_common\_cov = 1  
##   
## Computational engine: klaR

loans\_lda\_wf <- workflow() %>%   
 add\_model(loans\_lda\_model) %>%   
 add\_recipe(loans\_recipe)  
loans\_lda\_wf

## ══ Workflow ════════════════════════════════════════════════════════════════════  
## Preprocessor: Recipe  
## Model: discrim\_regularized()  
##   
## ── Preprocessor ────────────────────────────────────────────────────────────────  
## 3 Recipe Steps  
##   
## • step\_YeoJohnson()  
## • step\_normalize()  
## • step\_dummy()  
##   
## ── Model ───────────────────────────────────────────────────────────────────────  
## Regularized Discriminant Model Specification (classification)  
##   
## Main Arguments:  
## frac\_common\_cov = 1  
##   
## Computational engine: klaR

loans\_last\_fit\_lda <- loans\_lda\_wf %>%   
 last\_fit(split = loans\_split)

## Warning: package 'klaR' was built under R version 4.2.2

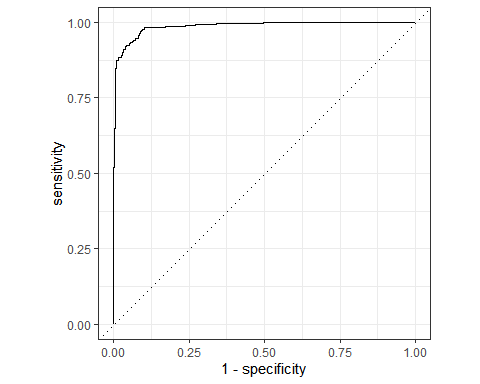
loans\_last\_fit\_lda %>% collect\_metrics()

## # A tibble: 2 × 4  
## .metric .estimator .estimate .config   
## <chr> <chr> <dbl> <chr>   
## 1 accuracy binary 0.944 Preprocessor1\_Model1  
## 2 roc\_auc binary 0.987 Preprocessor1\_Model1

loans\_lda\_predictions <- loans\_last\_fit\_lda %>%   
 collect\_predictions()  
  
loans\_lda\_predictions

## # A tibble: 1,028 × 7  
## id .pred\_yes .pred\_no .row .pred\_class loan\_default .config   
## <chr> <dbl> <dbl> <int> <fct> <fct> <chr>   
## 1 train/test split 0.299 0.701 4 no yes Preproce…  
## 2 train/test split 0.653 0.347 17 yes no Preproce…  
## 3 train/test split 0.000287 1.00 19 no no Preproce…  
## 4 train/test split 0.00712 0.993 22 no no Preproce…  
## 5 train/test split 0.998 0.00181 41 yes yes Preproce…  
## 6 train/test split 0.000270 1.00 46 no no Preproce…  
## 7 train/test split 0.957 0.0432 48 yes yes Preproce…  
## 8 train/test split 0.430 0.570 49 no yes Preproce…  
## 9 train/test split 1.00 0.0000880 60 yes yes Preproce…  
## 10 train/test split 0.000136 1.00 70 no no Preproce…  
## # … with 1,018 more rows

loans\_lda\_predictions %>%   
 roc\_curve(truth = loan\_default, estimate = .pred\_yes) %>%   
 autoplot()



conf\_mat(loans\_lda\_predictions, truth = loan\_default, estimate = .pred\_class)

## Truth  
## Prediction yes no  
## yes 348 23  
## no 35 622

f\_meas(loans\_lda\_predictions, truth = loan\_default, estimate = .pred\_class)

## # A tibble: 1 × 3  
## .metric .estimator .estimate  
## <chr> <chr> <dbl>  
## 1 f\_meas binary 0.923

— End of the Project —