Machine Learning With Tidymodels

Hamid Abdulsalam

2nd October 2021

Programme Outline

1. **Introduction to Machine Learning**
2. **A brief Introduction to Machine Learning Models**
3. **The Tidymodels Package**
4. **Case Study : Payfrax Bank**
5. **Conclusion**

About myself

About myself

- Data Scientist at Factual Analytics, Lagos
- M.Sc in Statistics (ABU, Zaria)
- B.Sc in Statistics (UNN, Nsukka)
- Research Interests include statistical inference & computing, ML & DL

Contact:

- hamid@factual.ng

Introduction to Machine Learning?

What is Machine Learning?

Machine Learning is a field of study at the intersection of Statistics and Computer Science.It focuses on creating algorithms that can be applied to datasets in order to detect and exploit patterns.

The primary use case for Machine Learning in business is to support decision making, in which data scientists attempt to discover and exploit patterns in datasets to achieve a business or research goal.

what is Machine Learning Contd?

Many Machine Learning models were developed from statistical learning, others were developed independently. In statistics, model interpretability is always our major concern. Hence, most statistical models can be explained. In contrast, machine learning is primarily concerned with the usefulness - or prediction accuracy - of the models utilized.

In a machine learning , it is often common to make use of a “black box” model whose internals are not fully known. In a nutshel, Machine Learning Models focus more on predictive accuracy over explanatory power.

what is Machine Learning (Contd)?

Machine Learning is frequently used to tackle two kinds of problems:

Regression Problem In a regression , we are trying to predict a numeric dependent variable. For example, we could be interested in predicting the credit score of certain consumers.

Classification Problem In a Classification task, we are trying to predict a categorical dependent variable. For example, we could be interested in predicting the loan status of certain consumer (Default or Not).

Types of Machine Learning Models

Supervised :In a supervised learning task, we try to predict the dependent variable using the independent variables in our dataset. For example, we may be interested in forecasting the likelihood that a bank client will fail on a loan based on his/her transactional data.

Unsupervised We don't have a target variable in an unsupervised learning problem. Our goal in unsupervised learning is to discover the hidden patterns in our dataset so that we can group similar observations into categories. For instance, we may be interested in segmenting customers based on their buying behaviour. Clustering techniques is an example of Unsupervised Machine Learning.

Types of Machine Learning Models (Contd)

Semi- Supervised : To create improved predictions on fresh observations, semi-supervised learning combines labeled and unlabeled data. Semi-supervised machine learning is not widely used by data scientists.

Reinforcement Learning : In reinforcement learning, an algorithm is created to learn a "policy" for interacting with an environment in such a way that it maximizes a predefined "objective." Reinforcement learning was used to build AlphaGo. AlphaGo is the first computer program to defeat a professional human Go player, as well as the first to defeat a Go world champion. It is widely regarded as the strongest Go player in history.

Machine Learning Models

Machine Learning Models

There are numerous machine learning models out there that can be used in performing machine learning tasks. In this training session, we will be looking at three ML models. They are :

Logistic Regression

Decision Tree

Random Forest

Let's briefly talk about these models

Logistic Regression

Logistic Regression is used to fit a curve to data in which the dependent variable is binary, or dichotomous. A logistic regression model allows us to establish a relationship between a binary outcome variable and a group of predictor variables. Why Logistic regression and NOT Linear regression?

Using linear model to explain a binary response will lead to model that performs below expectation.

Logistic Regression

The probability form of the Logistic regression model is given as:

$$\pi = \frac{e^{\beta_o + \beta_1 X}}{1 + e^{\beta_o + \beta_1 X}}$$

$\pi = Probability\ of\ the\ Event$

$\beta_0 = Intercept$

$\beta_1 = \ Slope$

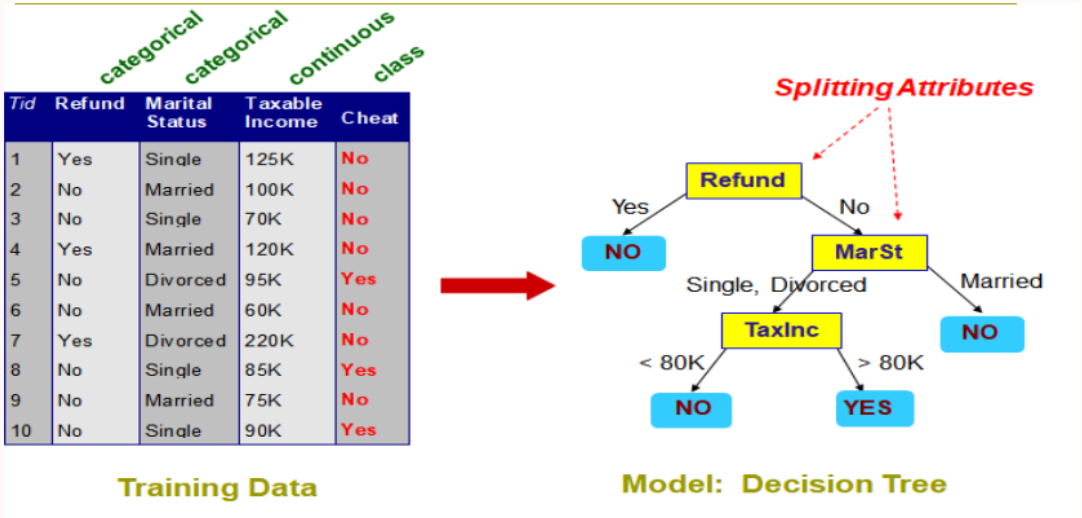
$X = Independent\ Variable$

Decision Tree

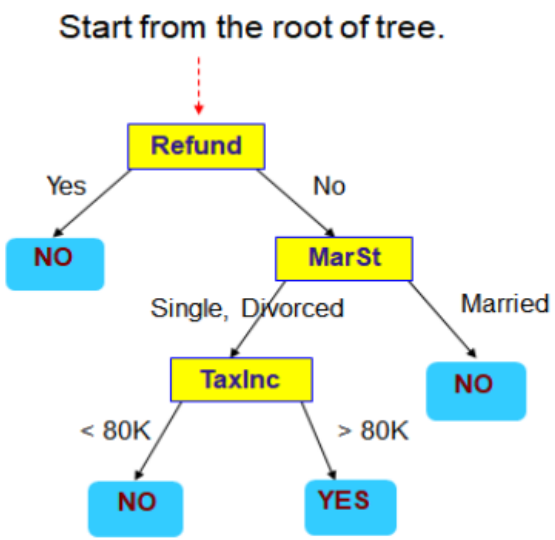
A decision tree is similar to a flowchart in that it is a relatively simple approach for making choices by repeatedly splitting the data. Humans can understand decision trees because they reflect rules. At each node of the decision tree, Gini impurity or information gain is used to pick an attribute to test.

information gain measures how well a given attribute separates the training examples according to their target classification. It is used to measure is used to select among the candidate attributes at each step while growing the tree.

Example of Decision Tree



Application of Decision Tree on a Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Random Forest

The decision tree's main drawback is that it tends to overfit the data. To compensate for the decision tree weakness, the random forest method employs a technique known as bagging.

In Random Forest, we usually create a new data from the original dataset to fit our algorithm. The resampling procedure of a random forest model is as follows:

1: A subset of the features is selected. The default is square root of the available features. If you have 16 features, each subset would choose 4 features at random.

2: We boostrap the data by resampling the rows with replacement. Some rows will be repeated and some will be missing in the new data.

Model Performance

To measure the performance of a regression issue, utilize the root mean squared error (RMSE), which is what linear regression (lm in R) aims to minimize. Model prediction accuracy will be used for categorization. Typically, we will not want to construct any of these metrics using data that were also used to generate the model. As a result, we must separate the data into Training and Test data. The training set is used to create the model, while the test set is used to evaluate model performance.

Regression

- MSE – Mean Squared Error
- RMSE – Root Mean Squared Error
- MAE – Mean Absolute Error
- R^2 – A measure that is related to MSE and is scaled between 0 and 1

Classification

- Accuracy
- Precision
- Recall
- AUC – Area Under the Curve

Confusion Matrix

A confusion matrix contains information about the model's actual and predicted classifications. The data in the matrix is commonly used to evaluate the performance of such model. Confusion Matrix is frequently used to assess the performance of a model with binary or multiclass responses.

		Predicted	
Actual		Positive	Negative
	Positive	True Positives	False Negatives
	Negative	False Positives	True Negatives

Evaluation Metrics

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FN+FP}$$

$$\text{Sensitivity/Recall} = \frac{TP}{TP+FN}$$

$$\text{Specificity} = \frac{TN}{FP+TN}$$

$$\text{Precision} = \frac{TP}{TP+FP}$$

The most commonly used metric is the accuracy which is obtained by dividing the total number of correct classifications by the total number of observations

Tidymodels Package

Tidymodels Package

The Tidymodels Package, like the Tidyverse, is a package that contains all of the necessary machine learning frameworks in a single package. It breaks down the functionality into smaller packages that each focus on a certain purpose. By using library(Tidymodels), we can load the essential packages in the Tidymodels package. Some of the core Tidymodels packages are

Parsnip: Provides a single interface to numerous machine learning algorithms.

Rsample: For splitting our data into Train and Test.

Recipes: For Data Processing.

Yardstick: For Model Performance.

dials: For model tuning.

In this training, we will discuss these packages

The Case Study

Before exploring the Tidymodels package, let's examine the case study.

You are a data scientist with PayPrax Bank's analytic team. During the coronavirus (COVID-19) epidemic, many people banking with PayPrax may find themselves in financial difficulty, unable to make loan or credit card payments. PayPrax Bank is currently looking for methods to assist its loan clients who have been impacted by the epidemic and its economic consequences. PayPrax would aim to develop a predictive algorithm to identify loan clients who are most likely to be in financial trouble.

The Case Study

PayPrax just completed a data gathering effort to track its clients' financial distress condition. PayPrax intends to utilize this information to conduct a targeted campaign to identify the most at-risk loan clients and discuss potential choices with them. This is because, by finding and providing acceptable choices for these customers, PayPrax may be able to reduce the projected loss from them as compared to the situation of failing to detect them and allowing them to default. PayPrax, on the other hand, would prefer to limit interaction with loan clients who are not in financial trouble in order to prevent giving superfluous benefits (which would reduce PayPrax's income).

Case Study

You have been given two datasets. The first dataset, titled “Campaign,” is the campaign’s processed data, which includes the financial distress status of 5,000 loan clients. You have been instructed to create an accurate machine learning model based on this data in order to identify loan clients who are most likely to be in financial difficulty. The second dataset is called “new data,” and it contains information on another 3,000 loan clients. This is the dataset that will be utilized to determine the consumers who will be contacted.

Data Dictionary

Variable name	Description
Internal data	
loan_status	Loan status. This is the data collected from the first data collection campaign. 0 = "Not distress", 1 = "Distress".
loan_amount	Loan amount
loan_purpose	Loan purpose
income	Income of the customer. For joint application, it is the income of the primary customer.
debt_to_income	Debt-to-Income ratio. For joint application, it is the income of the primary customer. It is a number that represents the customer's total monthly debt obligation divided by his/her total monthly income. For example: If your monthly debt payments are $1,000 and your gross monthly income is $5,000, your debt-to-income ratio is 20%.
emp_length	Employment length 1 = "< 2 years", 2 = "10 + years", 3 = "2 - 5 years", 4 = "6 - 9 years", 5 = "unemployed"
application_type	Application type – single or joint loan application
income_joint	The combined income of the customers
debt_to_income_joint	The combined debt-to-income ratio of the customers.
home_ownership	Home ownership status
region	Region of residence in Mulberry City
External credit report data	
credit_score	It is a number that is calculated by credit scoring methods using credit information reported about the customer by credit providers (e.g. lenders, banks and other financial institution). A customer with high credit score is expected to be less likely to incur negative credit events (e.g. late payment or default) than a customer with low credit score.

Data Dictionary

credit_accounts	Number of current active credit accounts (e.g. credit card, loan, home mortgage, car loan or other credit).
recent_inquiry	If there is any inquiry in the past 12 months. An inquiry is a check into the customer's credit report by a company or individual. It typically occurs when the customer has applied for a credit account. How inquiries can affect a credit score varies depending on the frequency and recentness of enquiries, the type of credit applied for, and the type of credit provider.
delinquent	When was the last delinquent? A delinquent occurs when the customer falls behind on making required monthly payment on credit accounts (e.g. loans or credit card). Being late by more than a month is considered delinquent. The delinquent status about the customer is reported to the credit bureaus by the customer's credit providers.
credit_utilization	It is a ratio of the amount of revolving credit the customer is currently using divided by the total amount of revolving credit he/she has available. Credit utilisation ratio is based solely on revolving credit (e.g. credit cards and lines of credit). It does not include installment loans like mortgage or car loans. For example, suppose you have a credit card and a line of credit, each with a limit of $5,000, for a total credit limit of $10,000. Let's say you owe $1,000 on the line of credit and $2,000 on the credit card, for a total of $3,000 owed. Given this, your credit utilization ratio is 30%.
past_bankrupt	Whether or not the customer has had a bankruptcy.

Importing the Data

```
##loading the necessary packages
library(gridExtra)
library(tidymodels)
library(tidyverse)
dt<-read.csv("campaign.csv", header = T,
             stringsAsFactors = T)
```

Glimpsing the Data

```
glimpse(dt)
```

```
## Rows: 5,000
## Columns: 18
## $ id                   <int> 491, 12656, 5580, 14775, 17992,
## $ loan_status          <int> 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0,
## $ loan_amount          <fct> MEDIUM, HIGH, HIGH, HIGH, HIGH,
## $ loan_purpose         <fct> wedding, debt_consolidation, deb
## $ income               <fct> MEDIUM, MEDIUM, MEDIUM, HIGH, ME
## $ debt_to_income       <fct> AVERAGE, HIGH, AVERAGE, AVERAGE,
## $ emp_length           <int> 2, 3, 1, 1, 2, 4, 1, 3, 4, 4, 3,
## $ application_type     <fct> SINGLE, SINGLE, SINGLE, SINGLE,
## $ income_joint         <fct> NA, NA, NA, NA, NA, NA, NA, NA,
## $ debt_to_income_joint <fct> NA, NA, NA, NA, NA, NA, NA, NA,
## $ home_ownership       <fct> RENT, OWN, RENT, OWN, RENT, OWN,
## $ region               <fct> NA, SOUTH, SOUTH, EAST, WEST, NA
## $ credit_score         <fct> HIGH, AVERAGE, AVERAGE, AVERAGE,
```

Factorising the Dependent variable and checking for Missing values

```
dt$loan_status<-as.factor(dt$loan_status)
levels(dt$loan_status)<-c("Not Distress", "Distress")
colSums(is.na(dt))
```

```
##                     id            loan_status             loan_amoun
##                      0                      0
##          loan_purpose                 income       debt_to_incom
##                      0                      0
##            emp_length     application_type          income_join
##                      0                      0                473
## debt_to_income_joint       home_ownership                regio
##                   4732                      0                103
##          credit_score      credit_accounts       recent_inquir
##                      0                      0
##            delinquent   credit_utilization          past_bankrup
##                      0                      0
```

Removing unwanted variables and Missing Values

```
dt$id<-NULL
dt$income_joint<-NULL
dt$debt_to_income_joint<- NULL
dt <- dt %>% drop_na()
```

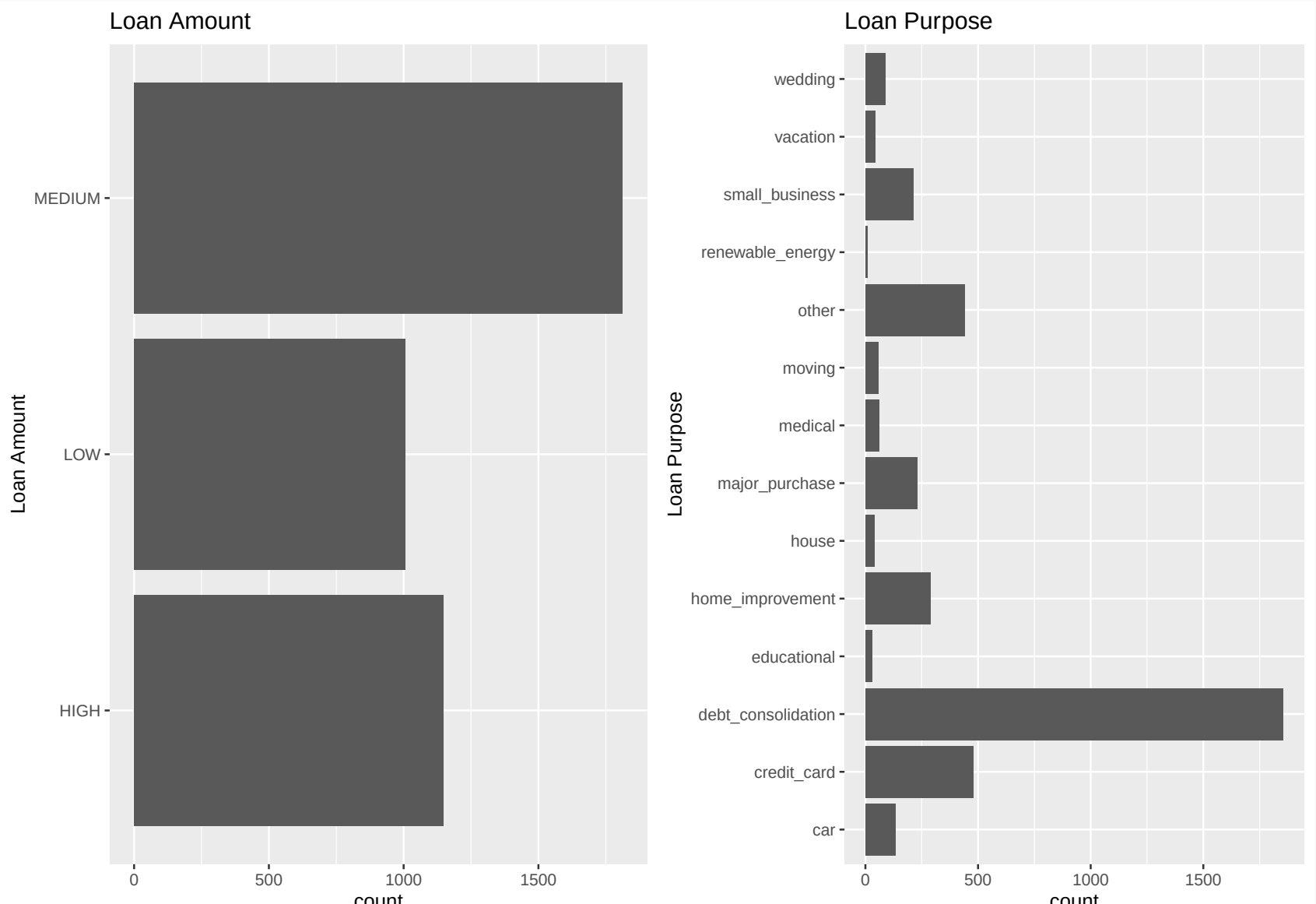
Data Exploration

Data exploratory is an important topic in data science. At this level, we are only interested in visualizing our data; we are not attempting to solve any problem.

```
p1 <- ggplot(dt, aes(x= loan_amount))+
ggtitle("Loan Amount") +
xlab("Loan Amount") + geom_bar()+ coord_flip()

p2 <- ggplot(dt, aes(x= loan_purpose)) +
ggtitle("Loan Purpose") +
xlab("Loan Purpose") + geom_bar()+coord_flip()
```

Data Exploraton

```
grid.arrange(p1, p2, ncol=2)
```

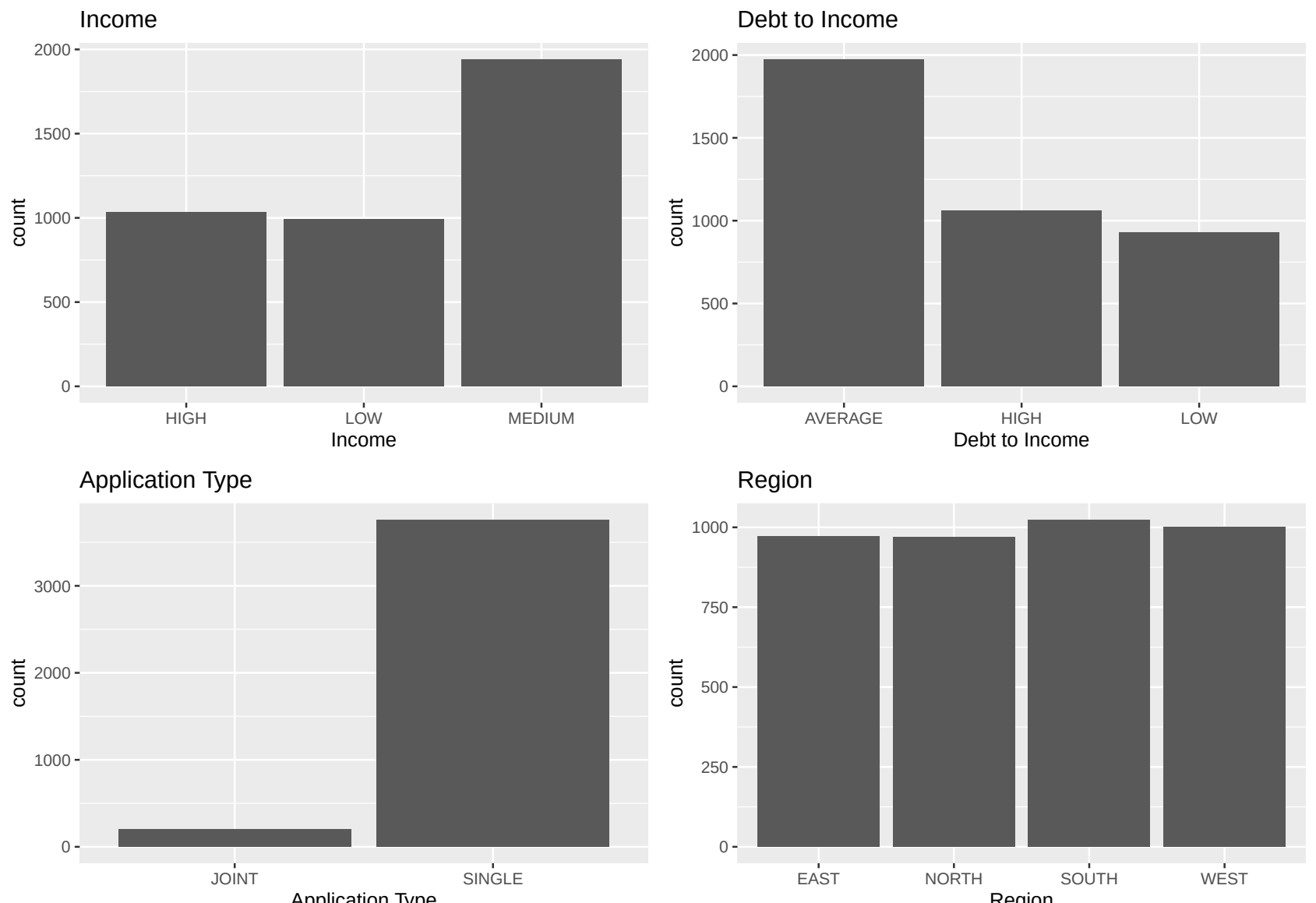


Data Exploration

```
p3 <- ggplot(dt, aes(x= income))+
ggtitle("Income") +
xlab("Income")+ geom_bar()

p4 <- ggplot(dt, aes(x= debt_to_income ))+
ggtitle("Debt to Income") +
xlab("Debt to Income") +
geom_bar()

p5 <- ggplot(dt, aes(x= application_type))+
ggtitle("Application Type")+
xlab("Application Type")+
geom_bar()

p6 <- ggplot(dt, aes(x= region))+
ggtitle("Region")+
xlab("Region") +
```

Data Exploration

```
grid.arrange(p3, p4, p5, p6,ncol=2)
```

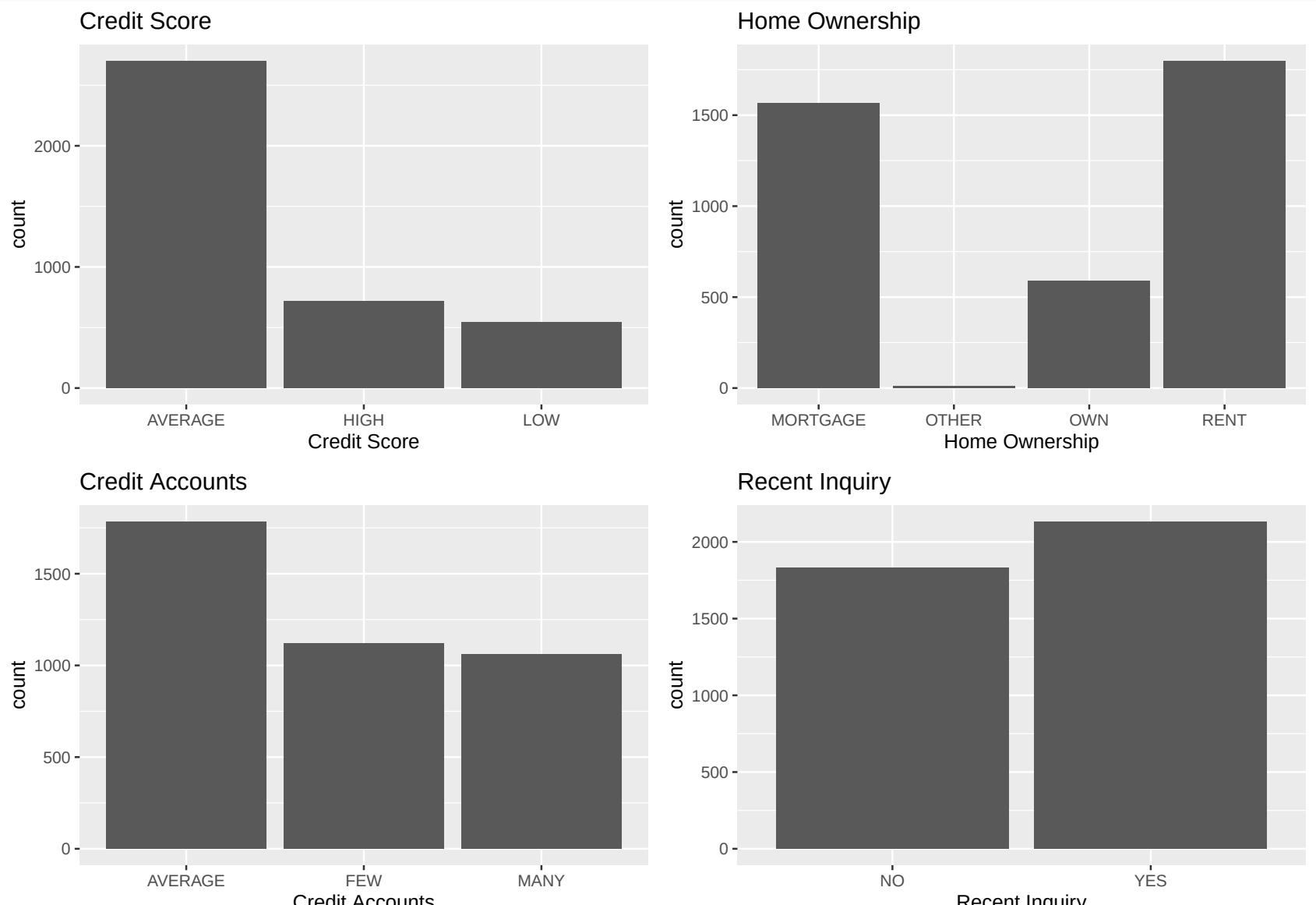


Data Exploration

```
p7 <- ggplot(dt, aes(x= credit_score )) +
ggtitle("Credit Score") +
xlab("Credit Score") +
geom_bar()

p8 <- ggplot(dt, aes(x= home_ownership )) +
ggtitle("Home Ownership") + xlab("Home Ownership")+
geom_bar()

p9 <- ggplot(dt, aes(x= credit_accounts)) +
ggtitle("Credit Accounts") +
xlab("Credit Accounts") +
geom_bar()

p10 <- ggplot(dt, aes(x= recent_inquiry )) +
ggtitle("Recent Inquiry") +
xlab("Recent Inquiry") +
```

Data Exploration

```
grid.arrange(p7, p8, p9, p10,ncol=2)
```



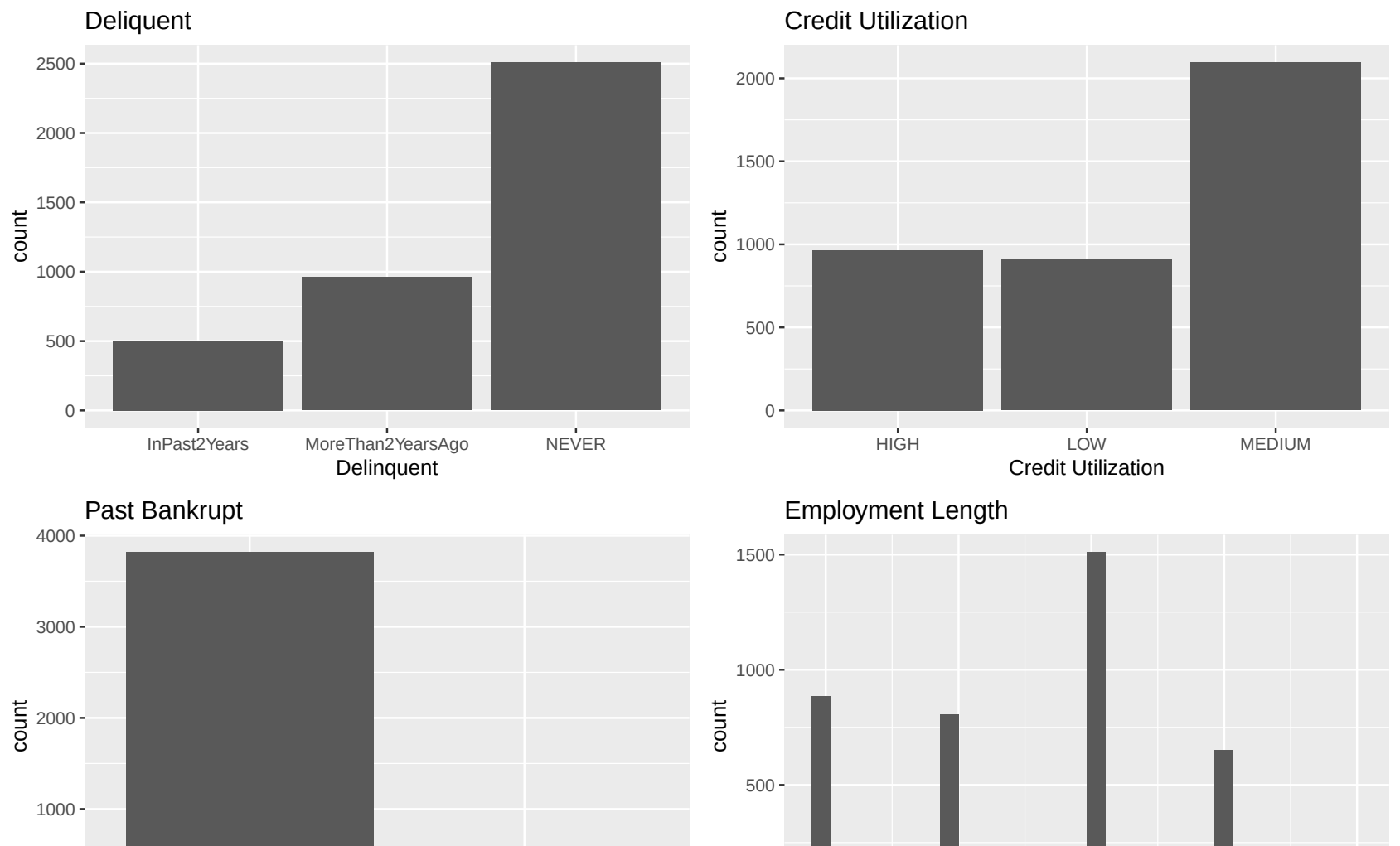
Data Exploration

```
p11 <- ggplot(dt, aes(x= delinquent )) +
ggtitle("Deliquent") +
xlab("Delinquent") +
geom_bar()

p12 <- ggplot(dt, aes(x= credit_utilization )) +
ggtitle("Credit Utilization") +
xlab("Credit Utilization") +
geom_bar()

p13 <- ggplot(dt, aes(x= past_bankrupt)) +
ggtitle("Past Bankrupt") +
xlab("Past Bankrupt") +
geom_bar()

p14 <- ggplot(dt, aes(x= emp_length)) +
ggtitle("Employment Length") +
```

Data Exploration

```
grid.arrange(p11, p12, p13, p14,ncol=2)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwi
```



Data Exploration

Loan Amount, Region, Credit Accounts and Recent Inquiry, Loan Amount, Income, and Debt to Income are all widely dispersed. The following variables are not widely distributed: Application Type, Home Ownership, Credit Score, and Loan Purpose. At the data processing step, we will need to transform these factor variables to dummy variables.

The minimum and maximum lengths of employment are one year and five years, respectively. The modal age is three years.

The Training and Testing Dataset

Let's start by splitting the data into Training and Test set using the **rsample** package in the tidymodels framework

```
set.seed(123)
dt_s <- initial_split(dt, prop = 0.70)
trainset <- training(dt_s)
testset <- testing(dt_s)
```

The "initial split" function will divide our data into a training set and a test set at random. We set the percentage that will be provided to the training set using the prop parameter. The "Strata" parameter ensures that the percentage of each level in the dependent variable remains constant across the training and test sets. We access the splits using the training and testing function.

Data Pre-Proceessing

Following the division of the dataset into Training and Test sets, the next step is to pre-process the data before modeling. Data pre-processing is frequently performed once our data has been divided. At this point, we will convert our category variables to dummy variables and center and scale the numeric variable. Because many machine learning algorithms operate best with dummy variables, it is critical to transform the category data to dummy variable. Before we can utilize them for modeling, we must explicitly encode them to numeric.

Data Pre-Processing

We will be using the **Recipe** Package in the tidymodels framework for our data pre-processing.

Using the recipe package, we often follow the cooking analogy. There are three major steps when pre-processing our data

recipe: We define the specification for the pre-processing steps. **prep:** The recipe is prepared on the training set **bake:** Apply the prepared recipe on the training and test set.

Data Pre-Processing

The Data Pre-processing step

```
rec_dt <- recipe(loan_status~., data = trainset)%>%
  step_dummy(all_nominal_predictors(), one_hot = T)%>%
  step_center(all_numeric_predictors())%>%
  step_scale(all_numeric_predictors())%>%
  prep(trainset)

bake_train<- bake(rec_dt, new_data = trainset)
bake_test<- bake(rec_dt, new_data = testset)
```

The specification is defined by the recipe. The step dummy converts nominal predictor variables to dummy variables, whereas the step center and scale centralize and scale numerical predictor variables. The Prep argument is prepared on the entire trainset's steps.

Modelling

After Pre-Processing our datasets, it's time to fit our models using the parsnip package in tidymodels. We will be fitting three Machine learning models; Logistic , Decision Tree and Random forest. Let's start with logistic regression

```
##Fitting logistic regression
logis<- logistic_reg()%>%
   set_engine("glm")%>%
   set_mode("classification")%>%
  fit(loan_status~., data=bake_train)
```

Here we fitted a logistic regression on our training data set, we started by creating the logistic regression model specification (logistic_reg), the we set the engine to be glm and mode to be classification, we finally fit the model by calling the fit function.

Modelling

Let's fit our Decision tree and Random Forest model.

```
##Fitting  decision tree
dtree<- decision_tree()%>%
   set_engine("rpart")%>%
   set_mode("classification")%>%
  fit(loan_status~., data=bake_train)

##Fitting  random Forest
rf<- rand_forest(mtry = 5, trees = 30)%>%
   set_engine("ranger",importance = "impurity")%>%
   set_mode("classification")%>%
  fit(loan_status~., data=bake_train)
```

Evaluating our model

We need to evaluate the performance of our model on the test data by using the fitted model to make predictions on our test dataset

```
log_pred <- bind_cols(
select(bake_test, loan_status),
predict(logis, bake_test)
)

dt_pred <- bind_cols(
select(bake_test, loan_status),
predict(dtree, bake_test)
)

rf_pred <- bind_cols(
select(bake_test, loan_status),
predict(rf, bake_test)
)
```

Evaluating our Model

Logistic Regression Performance

```
metrics(log_pred, truth = loan_status, estimate = .pred_class)
```

```
## # A tibble: 2 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.646
## 2 kap      binary         0.121
```

Decision Tree Performance

```
metrics(dt_pred, truth = loan_status, estimate = .pred_class)
```

```
## # A tibble: 2 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.650
## 2 kap      binary         0.0804
```

Evaluating our model

Random Forest Performance

```
metrics(rf_pred, truth = loan_status, estimate = .pred_class)
```

```
## # A tibble: 2 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.639
## 2 kap      binary         0.105
```

Which of the models perform better?

As we can see from the performance of our model, random performed better than the other models, hence we will be recommending logistic regression as our final model.Let's examine the most important features in our model.

```
library(vip)
```

```
##
## Attaching package: 'vip'
```

```
## The following object is masked from 'package:utils':
##
##     vi
```

```
vip(rf)
```

Who are customers to be contacted?

If you remember vividly, PayPrax provided a second data set. This second dataset is the dataset that will be utilized to determine the consumers to be contacted by PayPrax.Using our recommended model, we will identify consumers that should contacted.

```
dt1<-read.csv("new_Data.csv", header = T, stringsAsFactors = T)
```

```
## Remove unnecessary variables and missing values
dt1$id<-NULL
dt1$income_joint<-NULL
dt1$debt_to_income_joint<- NULL
dt1 <- dt1 %>% drop_na()
```

```
bake_new<- bake(rec_dt, new_data = dt1)
```

Applying the Recommended Model on the New Data Set

```
new_pred <- predict(rf, bake_new)
dt2<-dt1%>%mutate(Loan_Status=new_pred$.pred_class)
table(dt2$Loan_Status)
```

```
##
## Not Distress     Distress
##         2103          309
```

The are 303 distressed customers in the new dataset that should be contacted.

That's it for today. Questions?