Loan Default Analysis

BUSN 41201 (Big Data) - Group 20

May 28, 2023

Content

- Executive Summary
- Introduction to the Data (OK)
  - Source (OK)
  - Parameters of the Data (OK)
- Exploratory Data Analysis (OK)
  - Data Cleaning
  - Data Visualization
- Questions that We Want to Solve
- Model Building
  - Regression
    - Logistics Regression on Default
    - LASSO
    - AIC
    - Principle Components Analysis
    - Compare the Result
  - Clustering
    - K Means
    - Hierarchical Clustering
  - Machine Learning
    - Decision Trees
    - Random Forest
    - Classification And Regression Trees (CART)
    - Neural Network
- Conclusion
- Potential Future Research
- Appendix

Executive Summary

Introduction to the Data

Source

Name: I-Cheng Yeh

Email addresses: (1) icyeh '@' chu.edu.tw (2) 140910 '@' mail.tku.edu.tw

Institutions: (1) Department of Information Management, Chung Hua University, Taiwan. (2) Department of Civil Engineering, Tamkang University, Taiwan. Other contact information: 886-2-26215656 ext. 3181

Parameters of the Data

This research employed a binary variable, default payment (Yes = 1, No = 0), as the response variable. This study reviewed the literature and used the following 23 variables as explanatory variables:

- X1: Amount of the given credit (NT dollar): it includes both the individual consumer credit and his/her family (supplementary) credit.
- X2: Gender (1 = male; 2 = female).
- X3: Education (1 = graduate school; 2 = university; 3 = high school; 4 = others).
- X4: Marital status (1 = married; 2 = single; 3 = others).
- X5: Age (year).
- X6 - X11: History of past payment. We tracked the past monthly payment records (from April to September, 2005) as follows:
  - X6 = the repayment status in September, 2005;
  - X7 = the repayment status in August, 2005;. . . ;
  - X11 = the repayment status in April, 2005. The measurement scale for the repayment status is: -1 = pay duly; 1 = payment delay for one month; 2 = payment delay for two months;. . . ; 8 = payment delay for eight months; 9 = payment delay for nine months and above.
- X12-X17: Amount of bill statement (NT dollar).
  - X12 = amount of bill statement in September, 2005;
  - X13 = amount of bill statement in August, 2005;. . . ;
  - X17 = amount of bill statement in April, 2005.
- X18-X23: Amount of previous payment (NT dollar).
  - X18 = amount paid in September, 2005;
  - X19 = amount paid in August, 2005;. . . ;
  - X23 = amount paid in April, 2005.

Exploratory Data Analysis

Data Cleaning

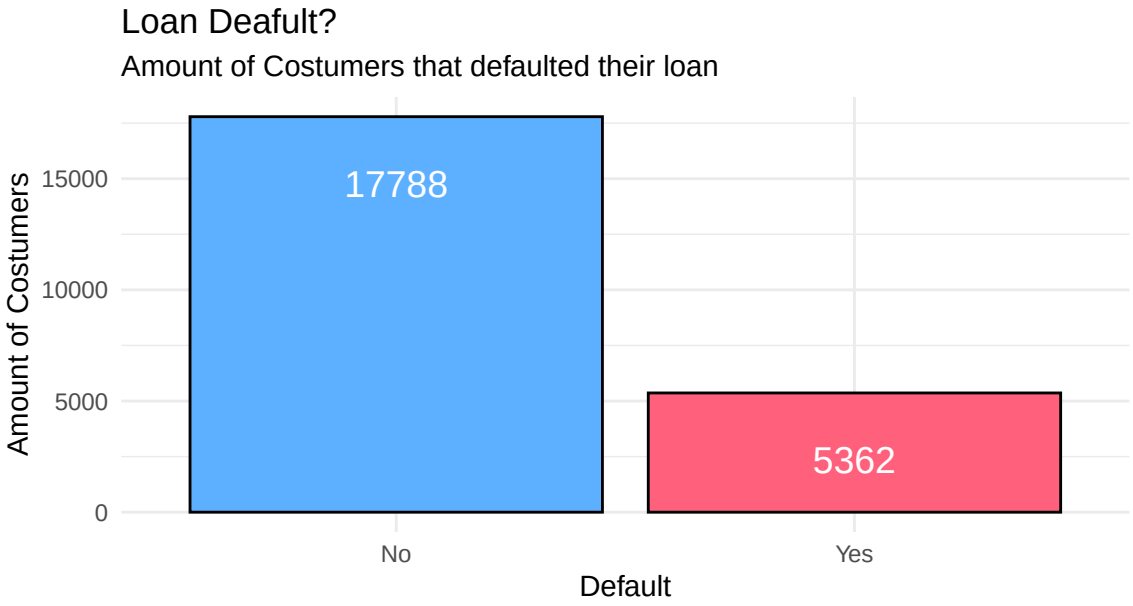
```
# Remove education indices which are not defined
data <- data %>% filter(EDUCATION<=4 & EDUCATION>0)
# Remove MARRIAGE indices which are not defined
data <- data %>% filter(MARRIAGE!=0)
for(i in 6:11){
  # -2 is not defined in the document
  data <- data[data[,i]!=-2,]
  # index pay duly as 0
  data[data[,i]==-1,i] <- 0
}
```

```
colnames(data)[24] <- 'default'

data <- data %>%
  mutate(SEX = factor(SEX, labels = c("Male", "Female")),
         EDUCATION = factor(EDUCATION),
         MARRIAGE = factor(MARRIAGE)) %>%
  mutate(EDUCATION = car::recode(EDUCATION, "c(0, 4, 5, 6) = 'Other';  1 = 'Grad.School'; 2 = 'Universi
         MARRIAGE = car::recode(MARRIAGE, "c(0, 3) = 'Other'; 1 = 'Married'; 2 = 'Single'"),
         default = factor(default, levels = c(0, 1), labels = c("No", "Yes")))

data <- data %>%
  mutate(PAY_0 = factor(PAY_0, order=TRUE,levels=c(seq(0,9, by = 1))),
         PAY_2 = factor(PAY_2, order=TRUE,levels=c(seq(0,9, by = 1))),
         PAY_3 = factor(PAY_3, order=TRUE,levels=c(seq(0,9, by = 1))),
         PAY_4 = factor(PAY_4, order=TRUE,levels=c(seq(0,9, by = 1))),
         PAY_5 = factor(PAY_5, order=TRUE,levels=c(seq(0,9, by = 1))),
         PAY_6 = factor(PAY_6, order=TRUE,levels=c(seq(0,9, by = 1))))
```

Data Visualization

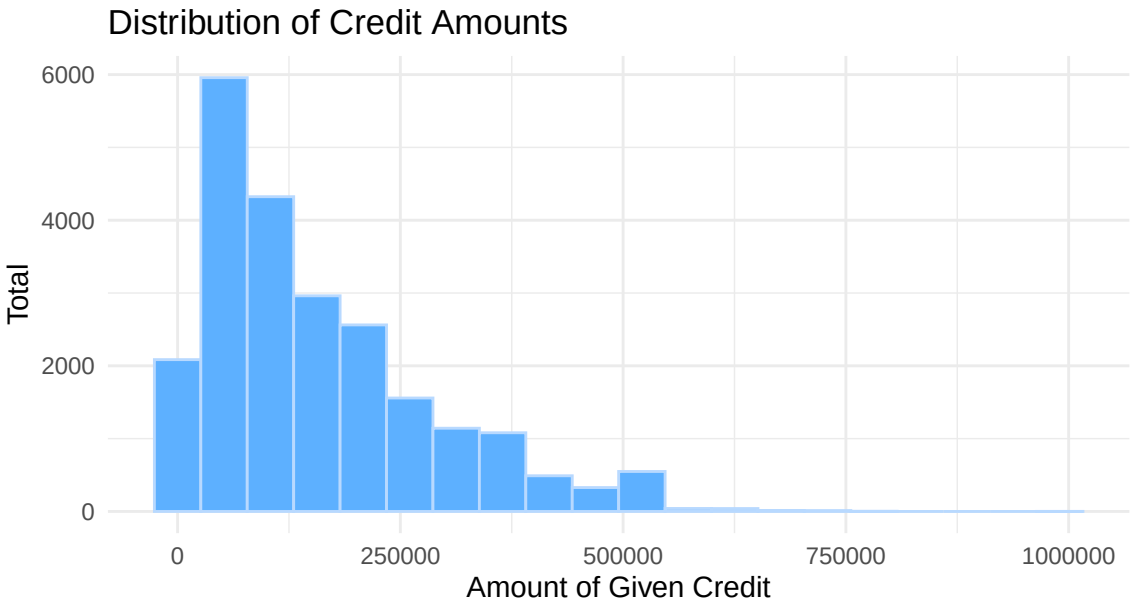
A. Default variable

The dataset consists of a total of 23150 individuals, out of which 17788 individuals do not default, and 5362 individuals default. Therefore, the overall default rate is 23.16%.



B. Amount of given credit (X1)

From the plot and table, it can be observed that the distribution appears to be right-skewed, indicating that there are relatively more individuals with lower credit amounts compared to higher credit amounts. Additionally, the distribution has a relatively large range, suggesting that the credit amounts vary significantly among the individuals in the dataset. Finally, we observe that the third quartile is $220,000, which implies that only a few people could get large amount of credit.



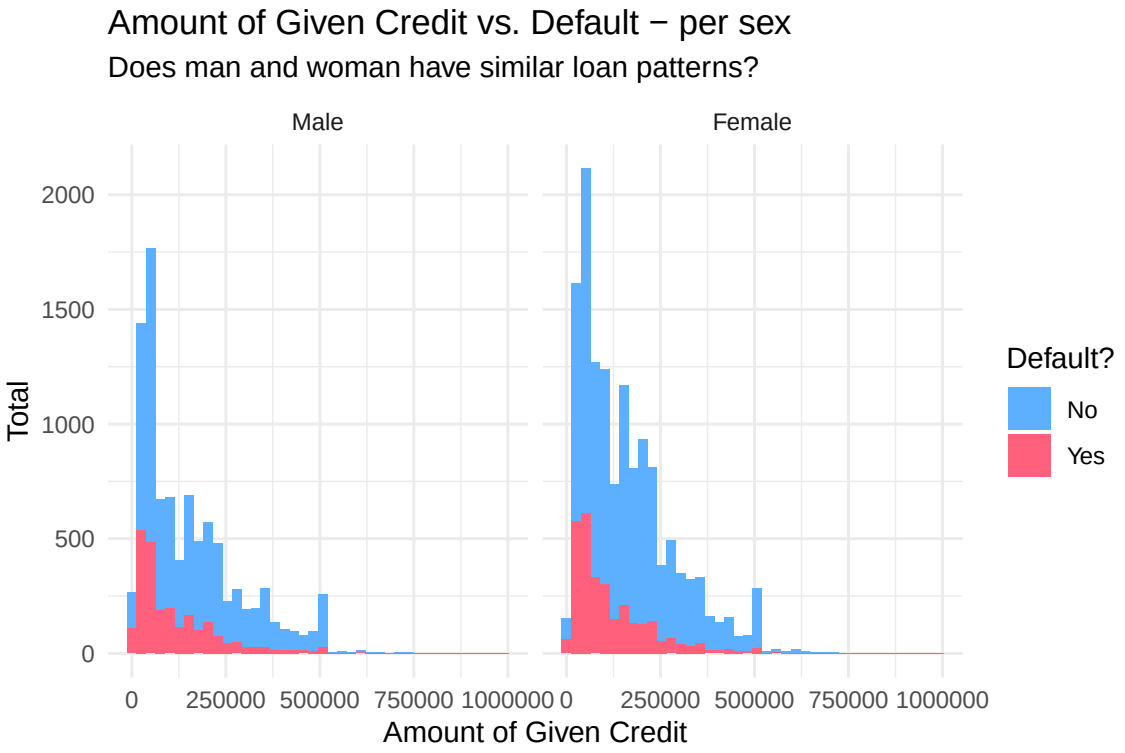
C. Gender

Among the male population of 9456 individuals, the default rate is 25.15%. In contrast, among the female population of 13694 individuals, the default rate is 21.81%.

Table 1: Loan Default Total among Gender

	Paid	Defaulted
Male	7078	2378
Female	10710	2984

From the figure, it can be observed that regardless of gender, the credit limits provided by the bank are concentrated between 100,000 and 200,000. Both male and female customers have their credit limits predominantly within this range. Also, the graph reveals that the default rate for males is slightly higher than that for females.



D. Education

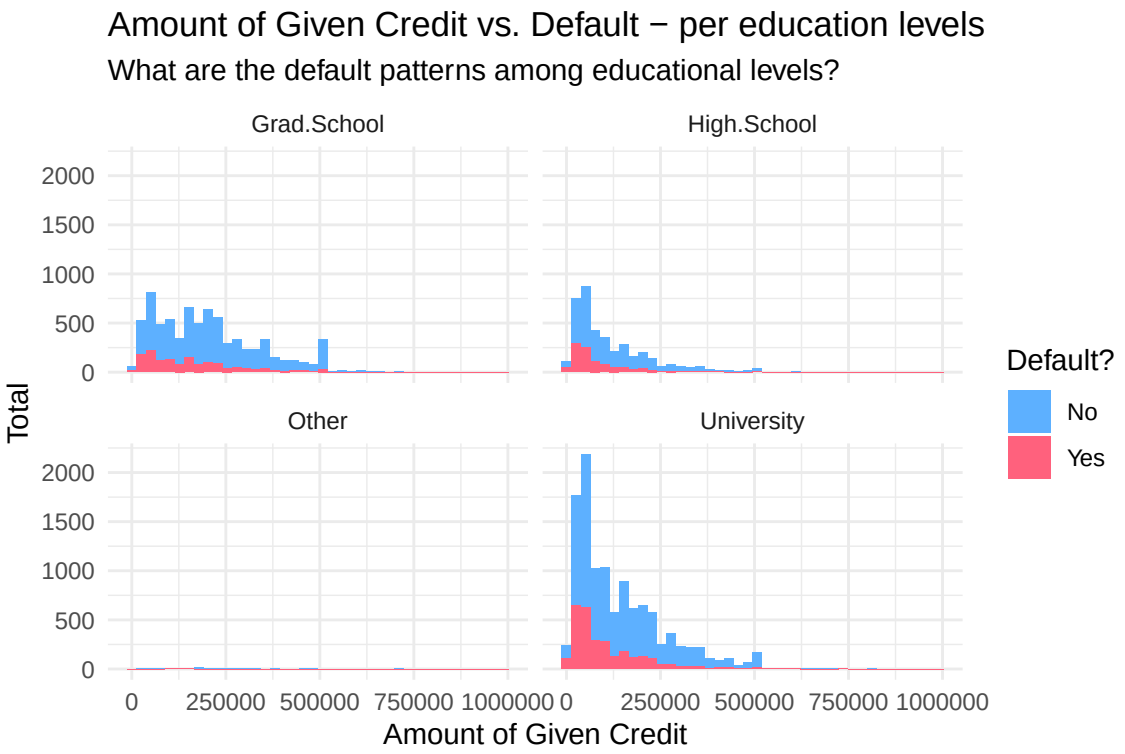
Among individuals with a high school education (3,995 people), the default rate is 25.86%. Among individuals with a university education (11,519 people), the default rate is 24.72%. Among individuals with a graduate school education (7,564 people), the default rate is 19.55%. Only 72 individuals fall under the "other" category, with a default rate of 4.17%.

Table 2: Loan Default Total among Educational Level

	Paid	Defaulted
Grad.School	6085	1479
High.School	2962	1033
Other	69	3
University	8672	2847

From the figure, it is evident that the majority of individuals have a university education, with a moderate default rate. The default rate is lowest for individuals with a graduate school education and highest for those with a high school education. This suggests a negative correlation between education level and default rates, indicating that individuals with higher education tend to have better financial capabilities, resulting in lower default rates. However, due to the small number of individuals in the "other" category and the uncertainty regarding their education level, no further explanation is provided for this group.

The figure reveals that customers with a high school or university education tend to be offered credit limits concentrated between 100,000 and 200,000. Specifically, there is a significant number of customers with a credit limit of 100,000. On the other hand, customers with a graduate school education have a relatively more evenly distributed credit limit range.



E. Marriage Status

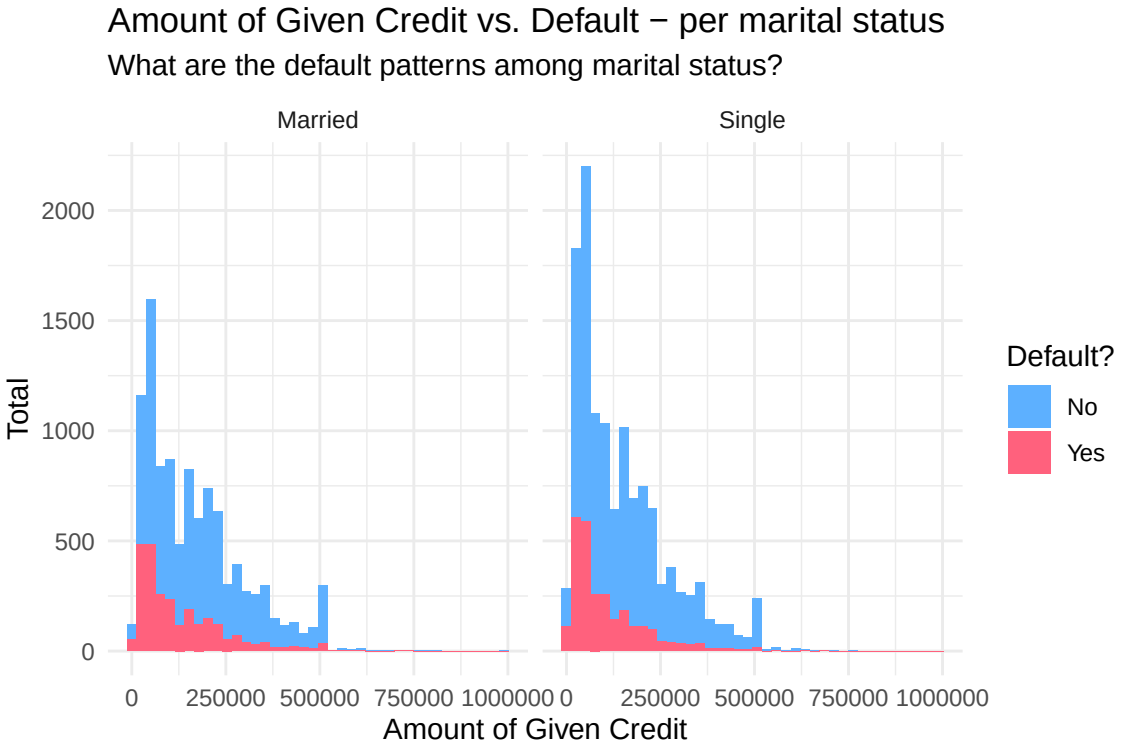
Among the married individuals, there are 10,354 people, with a default rate of 24.76%. Among the unmarried individuals, there are 12,527 people, with a default rate of 21.75%. There are 269 individuals classified under "other," with a default rate of 27.50%.

Table 3: Loan Default among Marital Status (%)

Married	Other	Single
24.76	27.51	21.75

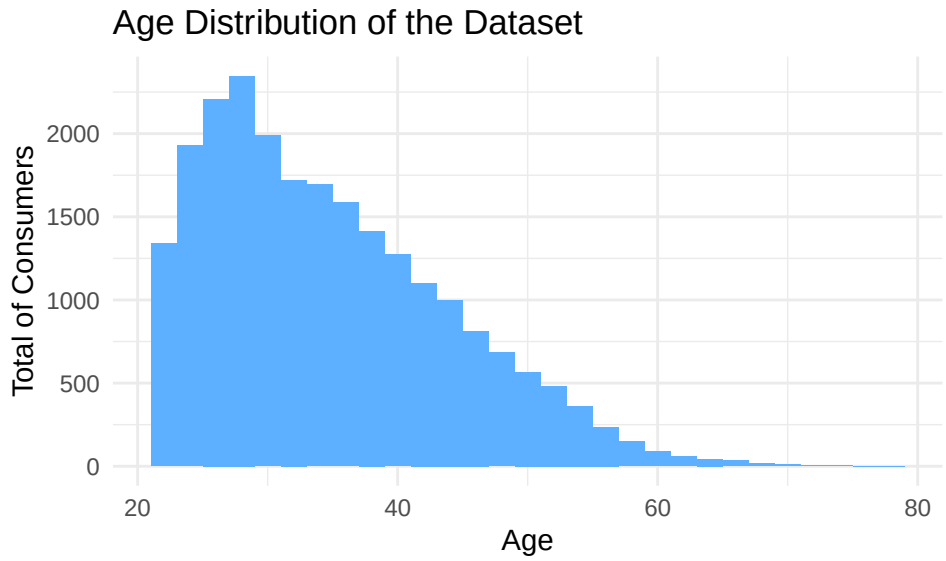
From the figure, it can be observed that the dataset contains a larger number of unmarried individuals, and their default rate is slightly lower compared to married individuals.

The figure indicates that regardless of marital status, the credit limits provided by the bank are concentrated between 100,000 and 200,000. Both married and unmarried individuals have their credit limits predominantly within this range.

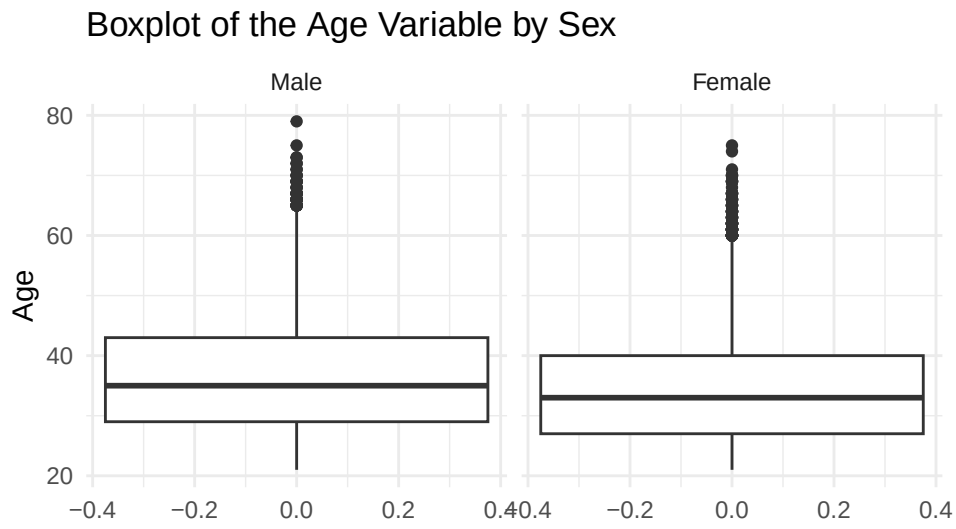


F. Age

From the figure, it is evident that the distribution exhibits a right-skewness. According to summary statistics of this variable, the youngest individual is only 21 years old, while the oldest individual is 79 years old. The average age is around 35 years, indicating that the majority of individuals fall into the youth and middle-aged categories



The boxplot can also helps us visualize the summary statistics of our data. It can be observed that the age distribution of males is slightly higher than that of females in the dataset.

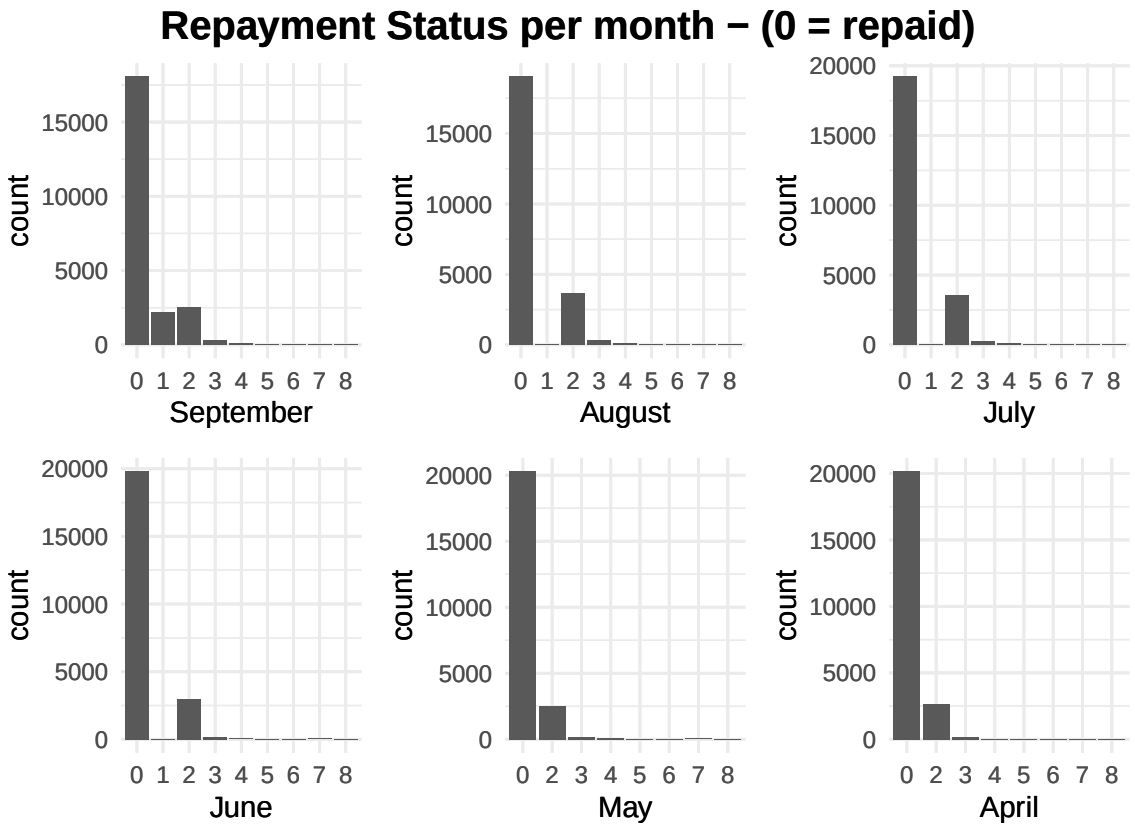


G. History of past-payments

From the figure, it can be observed that the majority of customers do not have any delayed payments. However, there is still a portion of customers who experience delayed payments, and it can be noted that these customers have a relatively higher default rate.

For customers who have no delayed payment for the past six months (PAY_0 ~ PAY_6 == 0), the default rate is 11.22%.

On the other hand, for customers who have delayed payments in every month for the past six months (PAY_0 ~ PAY_6 != 0), the default rate is 70.61%.



Questions that We Want to Solve

Model Building

Regression

Logistics Regression on Default

```
set.seed(122344)
# split data
idx <- sample(1:nrow(data), size = nrow(data) * 0.8, replace = FALSE)
train_data <- data[idx, ]
test_data <- data[-idx, ]
# initial value
logistic_fit <- glm(default ~ ., data = train_data, family = binomial)
summary_fit <- summary(logistic_fit)

significant_vars <- summary_fit$coefficients[summary_fit$coefficients[, "Pr(>|z|)"] < 0.05, ]

kable(significant_vars)
```

	Estimate	Std. Error	z value	Pr(>|z|)
LIMIT_BAL	-0.0000021	0.0000002	-9.188307	0.0000000
SEXFemale	-0.1576736	0.0409512	-3.850281	0.0001180
EDUCATIONHigh.School	-0.1292556	0.0642351	-2.012227	0.0441961
EDUCATIONOther	-1.3187153	0.6133143	-2.150146	0.0315437
MARRIAGESingle	-0.2293052	0.0464140	-4.940435	0.0000008
BILL_AMT1	-0.0000031	0.0000015	-2.045268	0.0408285
PAY_AMT1	-0.0000131	0.0000029	-4.469837	0.0000078
PAY_AMT2	-0.0000064	0.0000024	-2.654371	0.0079456

LASSO 1 (NOT RUNNING)

```
# library
library(glmnet)
# fit lasso
X = as.matrix(train_data[,1:23])
Y = train_data$default
cv_model = cv.glmnet(X, Y, family = "binomial", alpha = 1)
plot(cv_model) # mse for training models
best_lambda = cv_model$lambda.1se # choose lambda with 1 se
best_model = glmnet(X, Y, family = "binomial", alpha = 1, lambda = best_lambda)
coef(best_model)
# deviance
Y_predicted_in = predict(best_model, s = best_lambda, newx = X)
```

LASSO 2 (NOT RUNNING)

```
library(gamlr)
start.time = Sys.time()

x = data[,-(ncol(data))]
y = data$default

Lasso_model.1 = cv.gamlr(x, y=y, lambda.min.ratio=1e-3, family = "binomial")
plot(Lasso_model.1)

# We're using log-lambda around -7 for the Lasso model (based on AICc)

# R^2
Lasso.Rsq = summary(Lasso_model.1$gamlr)[which.min(AICc(Lasso_model.1$gamlr)),][4]
# Lasso probably not a good choice

end.time = Sys.time()
Lasso.time = end.time - start.time
```

AIC (?)

```
# step(logistic_fit)
```

Principle Components Analysis (NOT RUNNING)

```
library(corrplot)
x = data[,-(ncol(data))]
y = data$default
x_scale = scale(x)
corrplot(cor(x))
```

```
## Applying PC
pc_fx <- prcomp(x_scale)

## Rotating
v_fx <- predict(pc_fx)

## Exloring the data - principal component scores
fx2 <- pc_fx$rotation
fx2 <- apply(fx2,2,function(x){round(x,3)})

## Checking the values
summary(pc_fx)

## Observing the first components
fx2[,c(1:6)]

## Plotting our pcs
plot(pc_fx, main = "Scree Plot - International Currency Exchange Rates")
mtext(side=1, "Principal Components",  line=1, font=2)
```

From the output above, it seems that
PC1 captures the Amount of bill statement for the six month
PC2 captures the Repayment status for the six months
PC3 captures the Amount of previous payment
PC4 captures marriage and age which makes sense since this two covariates correlates positively as older people are more likely to be married.

```
# this is supplement to what I write above
plot(v_fx[, 1] ~ as.factor(data$LIMIT_BAL))
plot(v_fx[, 4] ~ as.factor(data$EDUCATION))
```

AICC and Lasso selection

AICC selects 19 which I don't know how to interpret. It seems that AICC thinks most PCs are important

```
df_v_fx <- as.data.frame(v_fx)

## Regressing default onto x - glm on first K
summary(PEglm <- glm(y ~ ., data = df_v_fx[,1:4]))

kfits <- lapply(1:23,
    function(K) glm(y ~ ., data = df_v_fx[,1:K,drop=FALSE]))

aicc <- sapply(kfits, AICc) # apply AICc to each fit
which.min(aicc) ## it likes 19 factors best

# Final
summary(PEglm <- glm(y ~ ., data = df_v_fx[,1:19]))
```

Here LASSO elinimates PC3-6, 10-12 and 19. I'm not sure how to interpret that

```
## Regressing y=default onto currency movement factors - Lasso
lassoPCR <- cv.gamlr(x=v_fx, y=y, nfold=20)

## Lasso
coef(lassoPCR)

## plot 'em
par(mfrow=c(1,2))
plot(aicc, pch=21, bg="maroon", xlab="K", ylab="AICc")

plot(lassoPCR)
```

LASSO PCR slightly better so should we conlcude PCA analysis didn't really contribute here?

```
### Running a lasso straight on the initial covariates
lasso <- cv.gamlr(x=x, y=y, nfold=20)

## Plotting
par(mfrow=c(1,2))
plot(lasso, main = "Regression onto original covariates")
plot(lassoPCR, main = "PCR")
```

```
## MSE values for both Lasso and PCR LASSO
print(lasso$lambda.min)
print(lassoPCR$lambda.min)
```

Classification

KNN (NOT RUNNING)

```
library(class)
trainRate = 0.8
rep = 1
K = 50
err = rep(0,K); errMat = matrix(0,rep,K);
for (r in 1:rep){
  trainNo = sample(dim(data)[1], trainRate*dim(data)[1])
  for (i in 1:K){
    knnfit = knn(train=data[trainNo,1:23],
                 test=data[-trainNo,1:23],
                 cl=data$default[trainNo],k=i)
    err[i] = 1-mean(knnfit == data[-trainNo,]$default)
  }
  errMat[r,] = err
}

plot(1:K,ylim=c(0,0.05),type="n",
     ylab="Testing error rate", xlab="K neighbors",
     main=paste("KNN", trainRate, "training data"))
points(1:K, colMeans(errMat),type="l",col=2,lwd=2)
segments(1:K, colMeans(errMat)-sqrt(diag(cov(errMat))),
         1:K, colMeans(errMat)+sqrt(diag(cov(errMat))),
         col="gray")
```

Linear Discriminant Analysis(LDA) (NOT RUNNING)

```
# library
library(MASS)
fit_lda = lda(train_data[,1:23],train_data$default)
attributes(fit_lda)
plot(fit_lda)
```

SVM (NOT RUNNING)

```
# library
# library(e1071)
# Train the SVM model
# svm_model <- svm(train_data[,1:23], train_data$default)
```

```
# Make predictions on the test set
# test_predictions <- predict(sum_model, newdata = test_data[, -c("default")])

# Evaluate the model's performance
# accuracy <- sum(test_predictions == test_data$default) / nrow(test_data)
```

Clustering

Cluster analysis is a valuable technique used in various fields to gain insights from data by identifying groups or clusters of similar objects or observations. For the Loan Default data, we have two main goals in clustering our data:

1. Segmentation and targeting: because clustering is popular in market segmentation and customer profiling, we want to identify the specific customer segment that are more likely to default their loans.
2. Anomaly detection: because clustering is a very good technique to identify outliers or anomalies within the data, this will be especially relevant in our case. For financial firms such ours (loan business) we need to identify the outlier that might have higher probability to get default, causing financial losses.

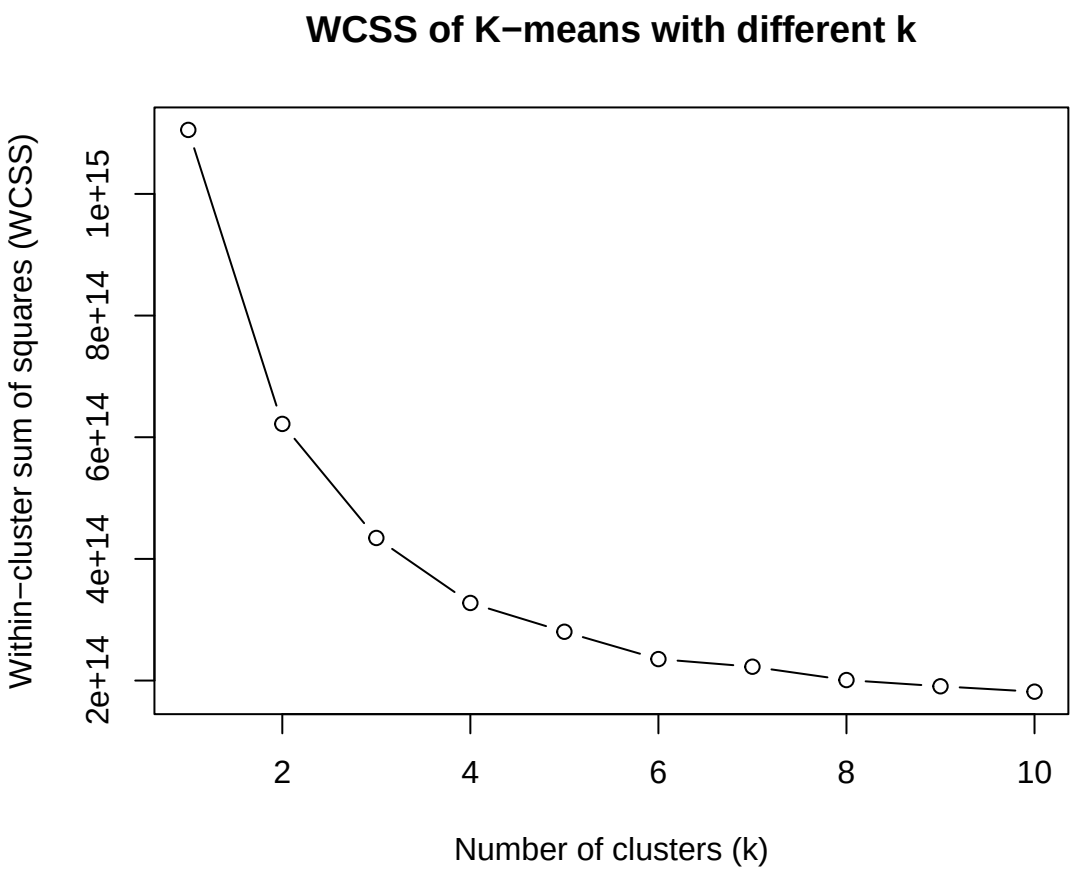
Following, we will perform three different clustering methods (K-means, Hierarchial clustering and Gaussian Mixture Models - GMM). Some methods are model-based, and some are not.

K Means

K-means is a popular clustering algorithm. Here, we aim to partition our dataset into a predetermined number of clusters. In the code bellow, we first choose the number of clusters (k) and randomly select k initial centroids. We then proceed to calculate the distance between each data point and the centroids. Assign each data point to the nearest centroid. Finally, we recalculate the centroids based on the data points assigned to each cluster, and repeat steps two and three until the clusters stabilize (convergence).

Once convergence is reached, the we then check the final cluster assignments and the coordinates of the centroid for each cluster, which will then be used for further analysis and interpretation.

We compare their within-cluster sum of square (WCSS) and choose the appropriate number k. Noted that K-means only works on numerical variables.




```
# find elbow, k=4 is appropriate
kmeans_result = kmeans(data_kmeans, centers = 5)
kmeans_result$centers
```

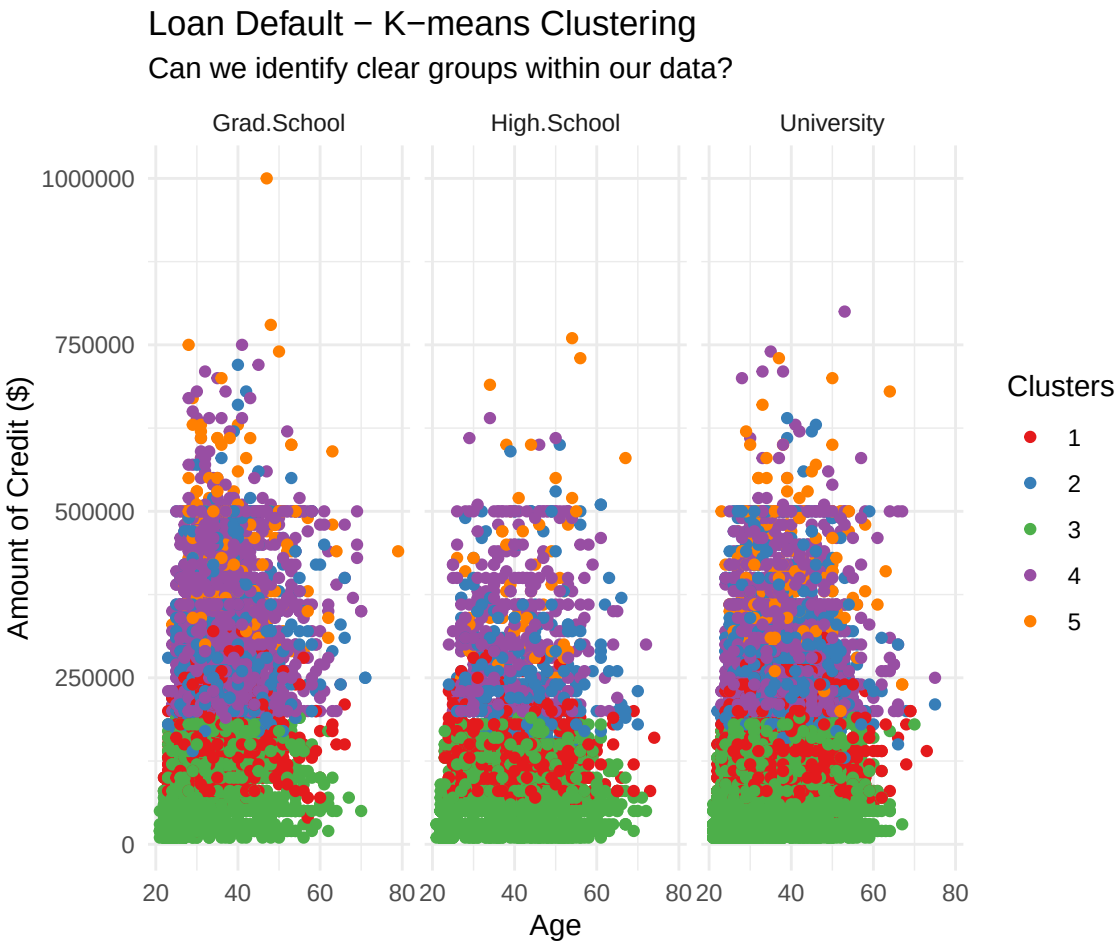
```
##   LIMIT_BAL      AGE BILL_AMT1 BILL_AMT2 BILL_AMT3 BILL_AMT4 BILL_AMT5
## 1 142554.41 35.22602  96795.80  94517.04  90067.02  83504.08  78541.07
## 2 275288.99 36.73119 190158.89 185627.41 178770.43 164922.87 152850.48
## 3  66732.18 34.35366  25222.09  24543.12  23313.48  21062.54  19541.61
## 4 313580.49 36.45758  23754.41  22092.16  21941.01  22108.82  21317.07
## 5 413509.33 38.54595 354819.22 350109.68 340912.72 320856.35 299168.37
##   BILL_AMT6  PAY_AMT1  PAY_AMT2  PAY_AMT3  PAY_AMT4  PAY_AMT5  PAY_AMT6
## 1  76110.90  6936.033  6403.116  5986.669  5594.856  5332.253  5173.778
## 2 145766.82 13893.089 14568.792 11346.958 10676.325 10142.867 10630.837
## 3  18839.88  2990.791  2864.303  2550.303  2410.688  2385.467  2364.063
## 4  20883.65  7741.274  8336.017  8413.509  7486.566  7802.061  8839.270
## 5 285641.60 20635.517 21725.942 20286.105 16612.510 17015.166 20258.841
```

When plotting the age as our x-axis and the amount of credit given as our y-axis, and breaking the plot by Education (into three categories), we have an interesting visualization of the job done by our clustering algorithm.

We can see that clusters three (3) and four (4) discovered a somewhat of a niche costumer. Cluster three seems to be formed predominantly by costumers for which a low limit was given, almost similar across all

ages. Cluster four (4), however, seems a costumer segment for which a higher amount of credit was given, which seems to be more common for people with grad school level of education.

Another relevant takeaways from our clusters is that, it seems that clusters one (1) and five (5) share a common characteristics across all level of education for a range of amount of credit given. This might indicate that, for these variables, this is a grey area and that we might need other models to help us identify which other variables can help us disentangle this information.



Hierarchical Clustering

Hierarchical clustering is a clustering algorithm that aims to create a hierarchical structure of clusters based on the similarity or dissimilarity between data points.

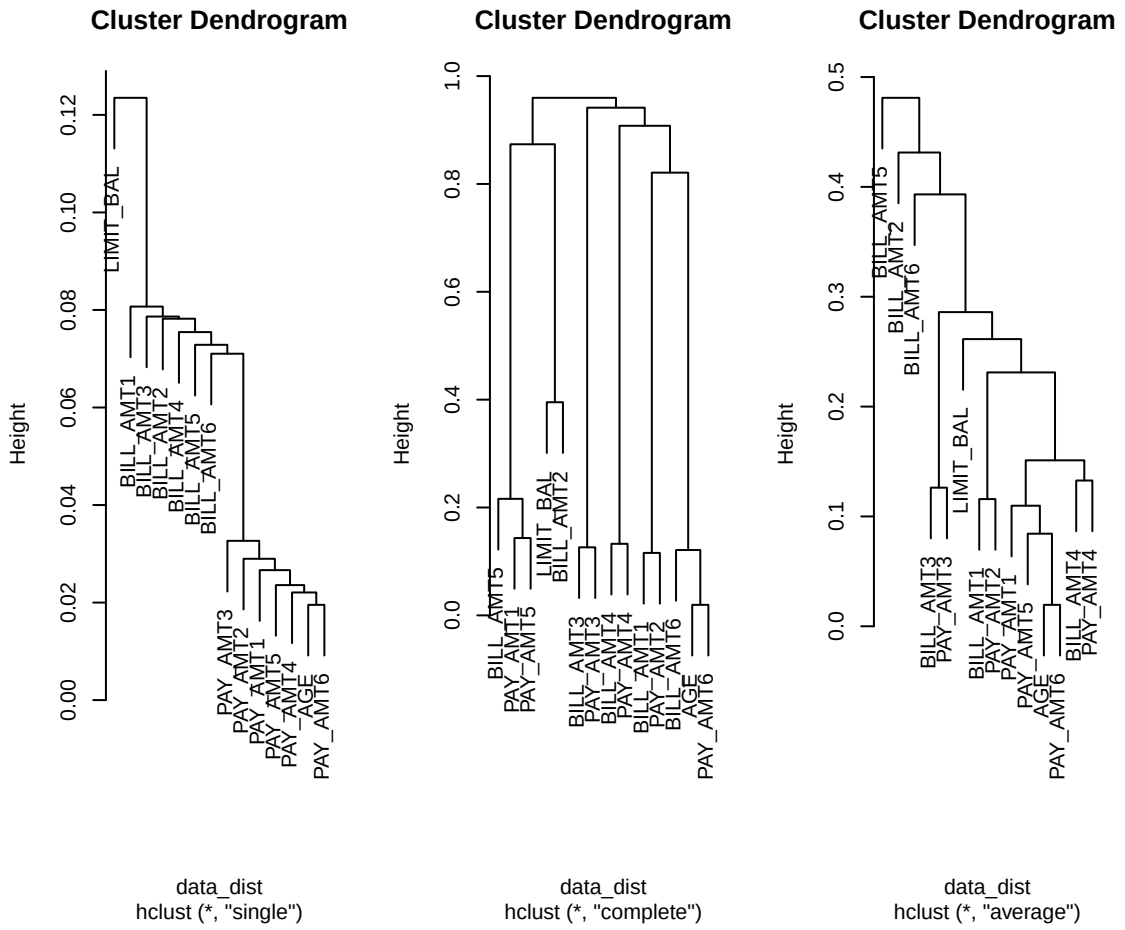
Hierarchical clustering use different linkage criteria to define the similarity or distance between clusters and guide the merging process. In our model, we will run our hierarchical clustering for the three following linkage criteria:

- Single linkage: The similarity between two clusters is defined as the minimum distance between any two data points, one from each cluster.
- Complete linkage: The similarity between two clusters is defined as the maximum distance between any two data points, one from each cluster.
- Average linkage: The similarity between two clusters is defined as the average distance between all pairs of data points, one from each cluster.

```
# variables clustering
# only use numerical data to do hierarchical clustering
# use correlation as similarity measure
# variable clustering
data_hc = data[,-c(2,3,4,6,7,8,9,10,11,24)]
cor_data = cor(data_hc)
data_dist = as.dist(cor_data)
Msingle = hclust(data_dist, method = "single")
Mcomplete = hclust(data_dist, method = "complete")
Maverage = hclust(data_dist, method = "average")
```

For each one of them, our findings can be summarized as follows:

1. Single Linkage: We can see that `PAY` and `BILL` two different categorical variables are clustering together. And, `Age` is the most closed to the most far payment `PAY_AMT6`.
2. Complete Linkage: It prefers to cluster some period of `PAY` and `BILL` together.
3. Average Linkage: The pattern is unclear. But, we still have that `Age` is the most closed to the most far payment `PAY_AMT6`.



By clustering variables, we can identify groups or clusters of variables that are related to each other. Variables within the same cluster often have similar behavior or share common underlying factors. This grouping can help in understanding the structure and organization of the dataset, revealing hidden patterns or relationships.

Next, we want to cluster by observations, but our observations is too large, that we need to reach from other approaches. I choose to randomly sample 500 observations from our dataset, and trying to find the pattern in the sample.

```
# Subset the data by randomly selecting 500 observations
subset_data <- data_hc[sample(nrow(data_hc), 500), ]

# Perform hierarchical clustering on the subset data
hc <- hclust(dist(subset_data))

# Summary of clustered sample observation
cluster_labels <- cutree(hc, k = 3)
subset_data$cluster_labels <- cluster_labels
summary(subset_data[subset_data$cluster_labels==1,])
summary(subset_data[subset_data$cluster_labels==2,])
summary(subset_data[subset_data$cluster_labels==3,])
```

Hierarchical clustering allows for a bottom-up approach, starting with individual data points and inter-

actively merging clusters based on similarity or distance. The resulting dendrogram provides a visual representation of the cluster hierarchy, and by cutting the dendrogram at an appropriate level, you can obtain clusters at different granularities.

This algorithm is flexible and can handle various types of data and distance/similarity measures, making it widely used in exploratory data analysis and visualization.

Gaussian Mixture Models (GMM)

Gaussian Mixture Models (GMM) is a probabilistic clustering algorithm that assumes the underlying data distribution is a mixture of Gaussian distributions. GMM clustering aims to model the data as a collection of Gaussian components, each representing a cluster.

But now we can use package `mclust` to avoid probability calculation. We also found three clusters in this method.

```
# Perform GMM clustering
data_mc <- data
gmm_model <- Mclust(data_mc[, -24], G = 3)
summary(gmm_model)

# Retrieve the cluster assignments
cluster_labels <- gmm_model$classification

# Summary of clustered sample observation
data_mc$cluster_labels <- cluster_labels
summary(data_mc[data_mc$cluster_labels==1,])
summary(data_mc[data_mc$cluster_labels==2,])
summary(data_mc[data_mc$cluster_labels==3,])

# Visualization
# plot(gmm_model, what=c("density"))
```

As a summary, here are the three main conclusions from our GMM model.

1. First cluster is a well-behaved group. Most of them pay the bill on time. They have the lowest default rate. We can proactively communicate with them, keeping customers informed about changes, updates, and new offerings. For this group, one major suggestion would be to increase the amount of relevant information shared, such as interest rate changes or new products/services. Being proactive in the our company's communication will likely demonstrate a transparency that keeps customers engaged.

2. Second cluster is more likely to be a group has higher default rate. We can develop informative content that emphasizes financial responsibility and highlights the importance of meeting financial obligations. Share tips on budgeting, managing debt, and improving credit scores. As a company trying to target loan default, providing a trusted advisor and resource for responsible financial practices can be useful for costumers identified within this group.

3. Third cluster includes some extreme outliers that has the highest payment. And, overall they have a little higher credit. We can offer exclusive benefits, discounts, or cashback programs that provide tangible value to your customers. For this group, establishing a formal channel of regular communications of rewards and benefits they can enjoy, is likely to reinforce their loyalty to our brand.

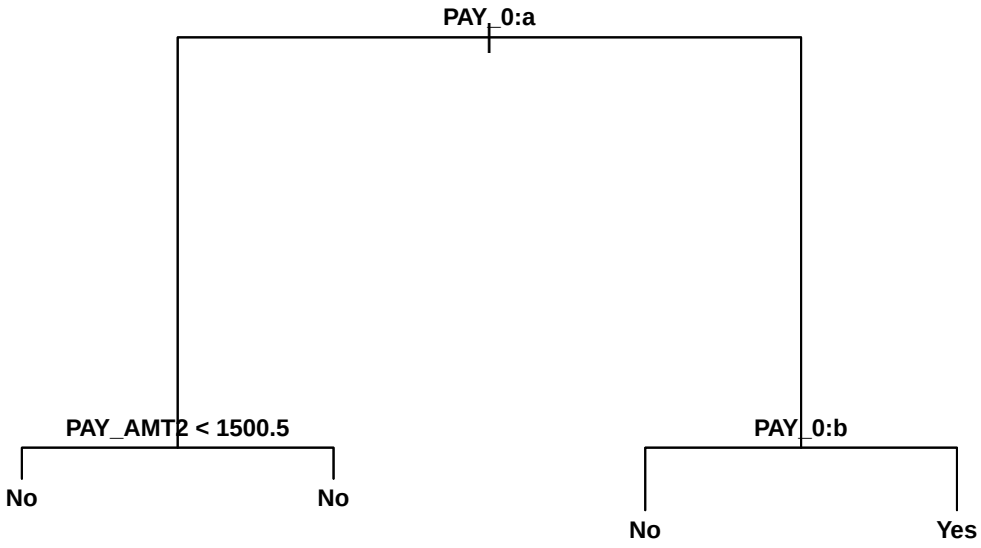
Machine Learning

Decision Trees

(CHUNK OF CODE NOT RUNNING)

```
library(tree)

start.time = Sys.time()

# for (i in c(2:4, 6:11)){
#   data[, i] = factor(data[,i])
# }

tree_model.1 = tree(factor(default) ~ ., data=data, mincut=1)

end.time = Sys.time()
tree_noncv.time = end.time - start.time

start.time = Sys.time()
tree_cvmodel = cv.tree(tree_model.1, K=10)
#data's dim is 23150, try to do as much as i can.(around 10%) just computational limitation
```

```
tree_model.2 = prune.tree(tree_model.1, best=4)

plot(tree_model.2)
text(tree_model.2, cex=.75, font=2)
```




```
tree_model.pred = predict(tree_model.2, data)
tree_model.pred_res = pred_res_func(tree_model.pred)
tree_model.accuracy = 1 - (length(which(tree_model.pred_res != data$default))/nrow(data))

end.time = Sys.time()
tree_cv.time = end.time - start.time
#:a is if Pay_0 = 0 or not
#:b is if Pay_0 = 1 or not
```

Random Forest

```
library(randomForest)
```

```
## randomForest 4.7-1.1

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:gridExtra':
##
##     combine
```

```
## The following object is masked from 'package:dplyr':
##
##     combine

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
rf_model = randomForest(factor(default) ~ ., data=data, ntree = 100, nodesize = 10) #ntree = 100(b/c co

rf_model.pred = predict(rf_model, data)
rf_model.rsq = 1 - (length(which(rf_model.pred != data$default))/nrow(data))
```

```
# - if we split into training and testing set
start.time = Sys.time()

train_idx = sample(1:nrow(data), 0.7 * nrow(data))  # 70% for training
train_data = data[train_idx, ]
test_data = data[-train_idx, ]

rf_spmodel = randomForest(factor(default) ~ ., data=train_data, ntree = 100, nodesize = 10)

rf_spmodel.pred = predict(rf_spmodel, test_data)
rf_spmodel.rsq = 1 - (length(which(rf_spmodel.pred != test_data$default))/nrow(test_data))
# it still looks good.

end.time = Sys.time()
rf.time = end.time - start.time
```

Artificial Neural Network (NOT RUNNING)

```
library(neuralnet)

start.time = Sys.time()

data_matrix = model.matrix(
  ~ LIMIT_BAL + SEX + EDUCATION + MARRIAGE + AGE + PAY_0 + PAY_2 + PAY_3 + PAY_4 + PAY_5 + PAY_6 + BILL_
  data = data)

train_data_matrix = data_matrix[train_idx, ]
test_data_matrix = data_matrix[-train_idx, ]

n = colnames(train_data_matrix[,-1])
f = as.formula(paste("factor(default) ~", paste(n[!n %in% "default"], collapse = " + ")))

nn_model = neuralnet(f,data=train_data_matrix, hidden=5, threshold=0.01,linear.output=T)
# Typically, increasing the number of hidden layers in a nn leads to improved accuracy. However, in thi

nn_model.pred = predict(nn_model, test_data_matrix)
nn_model.pred_res = pred_res_func(nn_model.pred)
nn_model.accuracy = 1 - (length(which(nn_model.pred_res != test_data$default))/nrow(test_data_matrix))
```

```
nn.time = end.time - start.time
```

Extra Analysis - model efficiency

The efficiency analysis of the different models reveals varying computational times. Given financial companies normally due with huge volumes of data, model efficiency can sooner become a very relevant topic. Thus, we displayed this table below to show a comparison of the time needed - in seconds - for each model to run.

The logistic regression model took approximately 1.20 seconds to run, indicating its relatively fast performance. The Lasso model exhibited a longer computation time of around 27.47 seconds, suggesting a higher complexity due to its regularization component. The tree model without cross-validation required about 7.13 seconds, while the tree model with cross-validation and the random forest model took around 95.63 seconds each, indicating a significantly longer computation time compared to the other models.Lastly, the neural network model also took approximately 95.63 seconds to complete, suggesting a longer computational duration due to its complex architecture.

```
# result_time = matrix(c(LR.time,
#                        Lasso.time,
#                        tree_noncv.time,
#                        tree_cv.time,
#                        rf.time,
#                        nn.time
#                        ), 1, 6, byrow = T)
#
# colnames(result_time) = c("Logistic Regression", "Lasso", "Tree(without cv)", "Tree(with cv)", "Rando
#
# kable(result_time)
```

Conclusion

Potential Future Research

Appendix