# 306 First Draft: Sam, Jangmin, Matt

2025-04-01

```
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(broom)
library(tibble)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v readr     2.1.5
## v lubridate 1.9.4     v stringr   1.5.1
## v purrr     1.0.4     v tidyr     1.3.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(pROC)
```

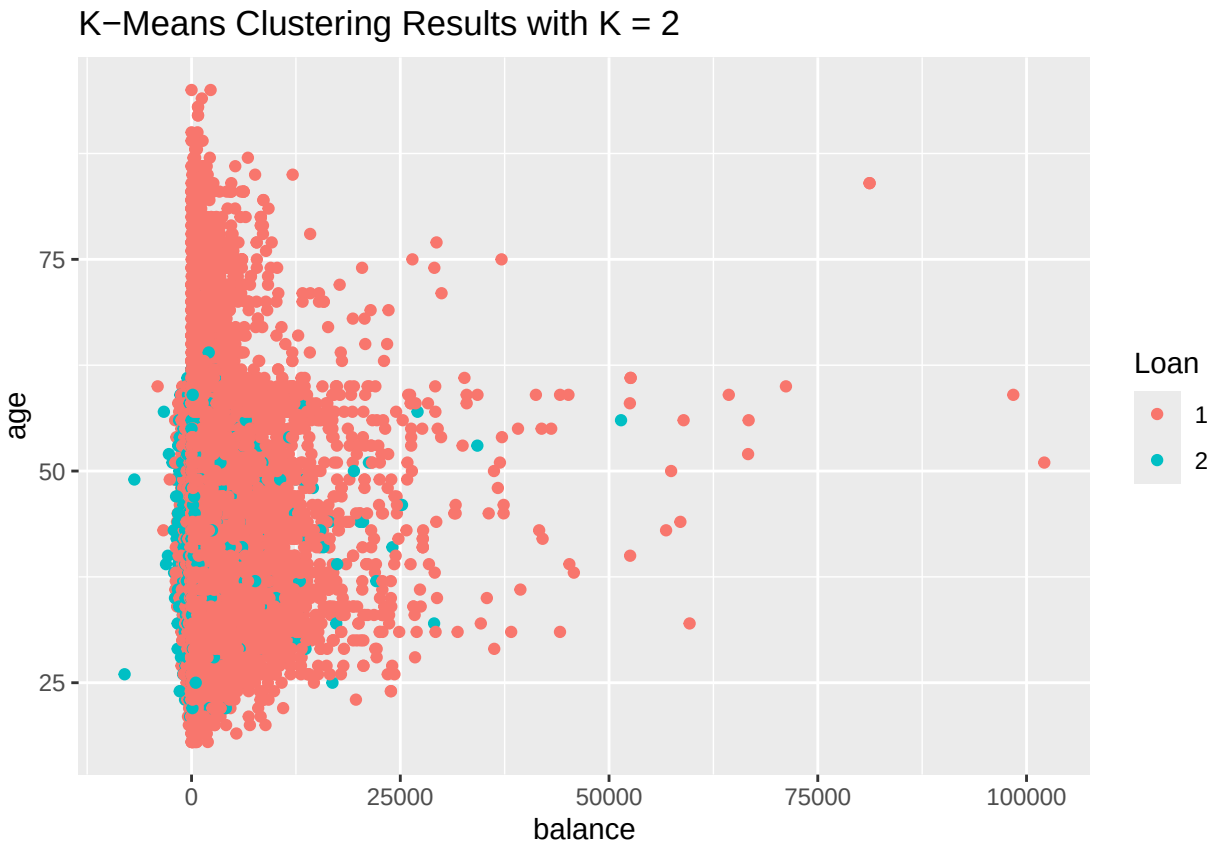
```
## Type 'citation("pROC")' for a citation.
##
## Attaching package: 'pROC'
##
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
bank_full <- read.csv("bank-full.csv", sep=";", header=TRUE)
bank_full <- bank_full |>
  select(age, job, marital, education, default, balance, housing, loan)
```

```r
bank_full$job <- as.numeric(factor(bank_full$job))
bank_full$marital <- as.numeric(factor(bank_full$marital))
bank_full$education <- as.numeric(factor(bank_full$education))
bank_full$default <- as.numeric(factor(bank_full$default))
bank_full$housing <- as.numeric(factor(bank_full$housing))
bank_full$loan <- as.numeric(factor(bank_full$loan))
```

```r
#bank_full_management <- bank_full |>
  #filter(job == 5)

#random_sample <- bank_full[sample(nrow(bank_full), 100), ]

set.seed(1)
bank_full_balance_scaled <- bank_full |>
  mutate(balance = scale(bank_full$balance)) |>
  mutate(age = scale(bank_full$age))

bank_full_scaled <- scale(bank_full)
bank_full_scaled <- bank_full_scaled[, -8]

km_out = kmeans(bank_full_balance_scaled, 2, nstart = 20)
km_clusters = km_out$cluster
#bank_full_scaled <- cbind(bank_full_scaled, enframe(bank_full$loan))
#bank_full_scaled <- bank_full_scaled[, -8]
#df = table(km_clusters, bank_full_balance_scaled$loan)



assignments <- augment(km_out, bank_full_balance_scaled)

#assignments$value <- as.character(assignments$value)
#ggplot(data = assignments, aes(x = balance, y = age, color = .cluster)) +
#  geom_point() + labs(color = "Cluster Assignment",
#                      title = "K-Means Clustering Results with K = 2",
#                      shape = "loan")
```

```r
bank_full$loan <- as.character(assignments$loan)

ggplot(data = bank_full, aes(x = balance, y = age, color = loan)) +
  geom_point() + labs(color = "Loan",
                      title = "K-Means Clustering Results with K = 2")
```




Clustering with education:

```
# bank_full_scaled <- scale(bank_full)
# bank_full_scaled <- bank_full_scaled[, -2]
#
# km_out = kmeans(bank_full_scaled, 12, nstart = 20)
# km_clusters = km_out$cluster
# bank_full_scaled <- cbind(bank_full_scaled, enframe(bank_full$job))
# bank_full_scaled <- bank_full_scaled[, -8]
# df = table(km_clusters, bank_full_scaled$value)
#
#
#
# assignments <- augment(km_out, bank_full_scaled)
#
# assignments$value <- as.character(assignments$value)
# ggplot(data = assignments, aes(x = balance, y = age, color = .cluster)) +
#   geom_point() + labs(color = "Cluster Assignment",
#                       title = "K-Means Clustering Results with K = 2",
#                       shape = "loan")
#
# ggplot(data = bank_full, aes(x = balance, y = age, color = job)) +
#   geom_point() + labs(color = "Job",
#                       title = "K-Means Clustering Results with K = 2")
#
# write.csv(df, "df.csv", row.names = FALSE)
```

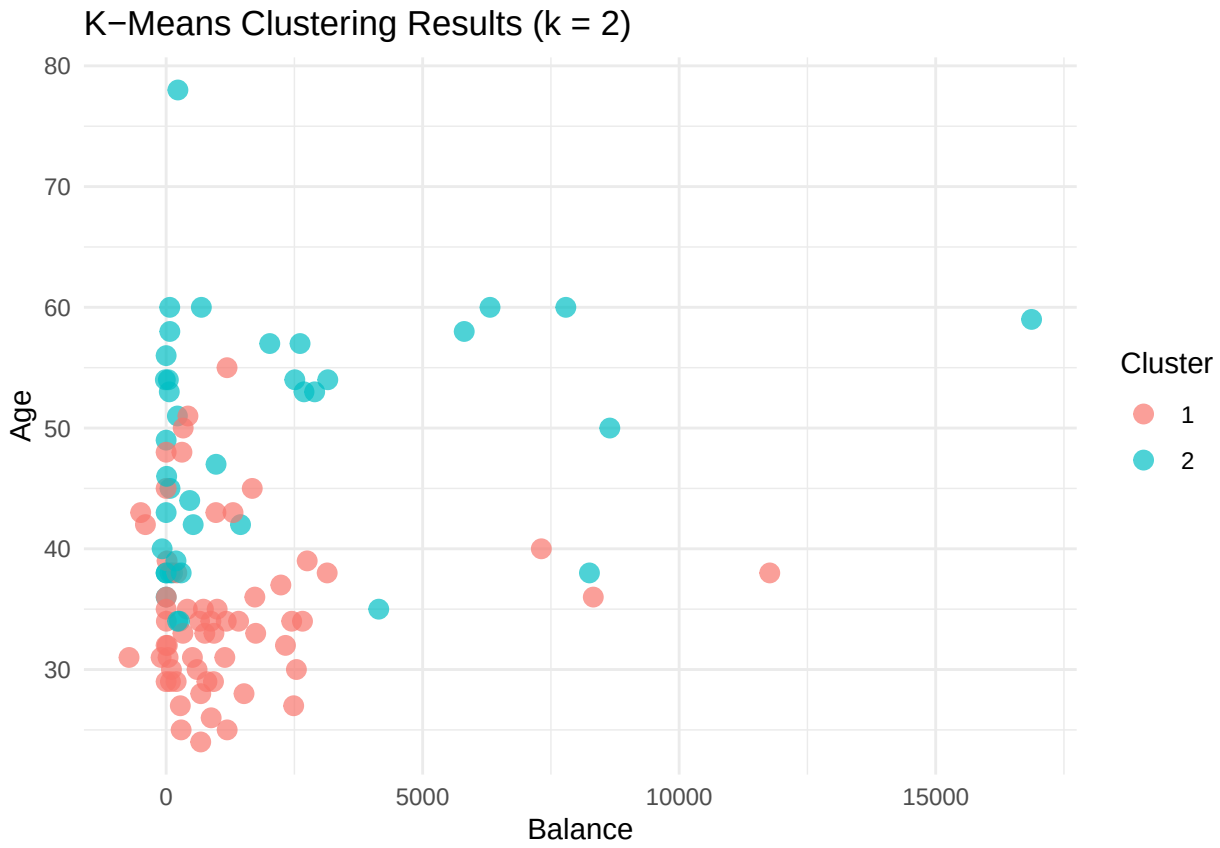
```r
# Load and select variables
bank_full <- read.csv("bank-full.csv", sep=";", header=TRUE)
bank_full <- bank_full |>
  select(age, job, marital, education, default, balance, housing, loan)

# Convert categorical variables to numeric
bank_full <- bank_full |>
  mutate(across(c(job, marital, education, default, housing, loan),
                ~ as.numeric(factor(.x))))
```

```r
# Optional: Take a random sample for speed
set.seed(1)
bank_sample <- bank_full[sample(nrow(bank_full), 100), ]

# Scale the numeric dataset before clustering
df_scaled <- scale(bank_sample)

# Run K-means clustering
km_out <- kmeans(df_scaled, centers = 2, nstart = 20)
#km_out <- kmeans(bank_sample, centers = 2, nstart = 20)
km_clusters <- km_out$cluster
table(km_clusters, bank_sample$loan)
```

```
##
## km_clusters  1  2
##           1 50 10
##           2 34  6
```

```r
# Add cluster assignments to dataset
assignments <- augment(km_out, bank_sample)
```

```r
# Visualize clusters
ggplot(assignments, aes(x = balance, y = age, color = .cluster)) +
  geom_point(size = 3, alpha = 0.7) +
  labs(
    title = "K-Means Clustering Results (k = 2)",
    x = "Balance",
    y = "Age",
    color = "Cluster"
  ) +
  theme_minimal()
```




```r
# Read the dataset and select predictors plus the target 'y'
bank <- read.csv("bank-full.csv", sep = ";", header = TRUE)
bank_class <- bank %>%
  select(job, marital, education, balance, y)

# Convert 'y' (loan status) to a factor with levels "no" and "yes"
# (Assuming your dataset uses these strings)
bank_class$y <- factor(bank_class$y, levels = c("no", "yes"))

# For simplicity in least squares classification, convert categorical predictors to numeric
bank_class$job <- as.numeric(factor(bank_class$job))
bank_class$marital <- as.numeric(factor(bank_class$marital))
bank_class$education <- as.numeric(factor(bank_class$education))

# Split the data into training (70%) and testing (30%) sets
set.seed(42)
train_index <- createDataPartition(bank_class$y, p = 0.7, list = FALSE)
train_data <- bank_class[train_index, ]
test_data  <- bank_class[-train_index, ]

# Convert the target to numeric for regression (e.g., no = 0, yes = 1)
train_data$y_num <- ifelse(train_data$y == "yes", 1, 0)
test_data$y_num  <- ifelse(test_data$y  == "yes", 1, 0)

# Fit a linear regression model (least squares classifier)
model_ls <- lm(y_num ~ job + marital + education + balance, data = train_data)
summary(model_ls)
```

```
##
## Call:
## lm(formula = y_num ~ job + marital + education + balance, data = train_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.64983 -0.13205 -0.11075 -0.08789  0.95324
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -9.461e-04  8.318e-03  -0.114    0.909
## job          3.219e-03  5.587e-04   5.763 8.34e-09 ***
## marital      2.091e-02  2.968e-03   7.043 1.92e-12 ***
## education    2.191e-02  2.458e-03   8.911  < 2e-16 ***
## balance      4.957e-06  5.919e-07   8.375  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.32 on 31644 degrees of freedom
## Multiple R-squared:  0.009006,   Adjusted R-squared:  0.008881
## F-statistic:  71.9 on 4 and 31644 DF,  p-value: < 2.2e-16
```

```r
# Predict on training and testing sets
y_hat_train <- predict(model_ls, newdata = train_data)
y_hat_test  <- predict(model_ls, newdata = test_data)

# Apply a decision boundary (here, 0.5) to determine class predictions
y_pred_train <- ifelse(y_hat_train >= 0.5, 1, 0)
y_pred_test  <- ifelse(y_hat_test  >= 0.5, 1, 0)

# Create confusion matrices for training and testing predictions
train_conf <- confusionMatrix(as.factor(y_pred_train), as.factor(train_data$y_num))
test_conf  <- confusionMatrix(as.factor(y_pred_test),  as.factor(test_data$y_num))
print(train_conf)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     0     1
##          0 27944  3703
##          1     2     0
##
##                Accuracy : 0.8829
##                  95% CI : (0.8793, 0.8865)
##     No Information Rate : 0.883
##     P-Value [Acc > NIR] : 0.5183
##
##                   Kappa : -1e-04
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9999
##             Specificity : 0.0000
##          Pos Pred Value : 0.8830
##          Neg Pred Value : 0.0000
```

```
##              Prevalence : 0.8830
##          Detection Rate : 0.8829
##    Detection Prevalence : 0.9999
##       Balanced Accuracy : 0.5000
##
##        'Positive' Class : 0
##
```

```
print(test_conf)
```

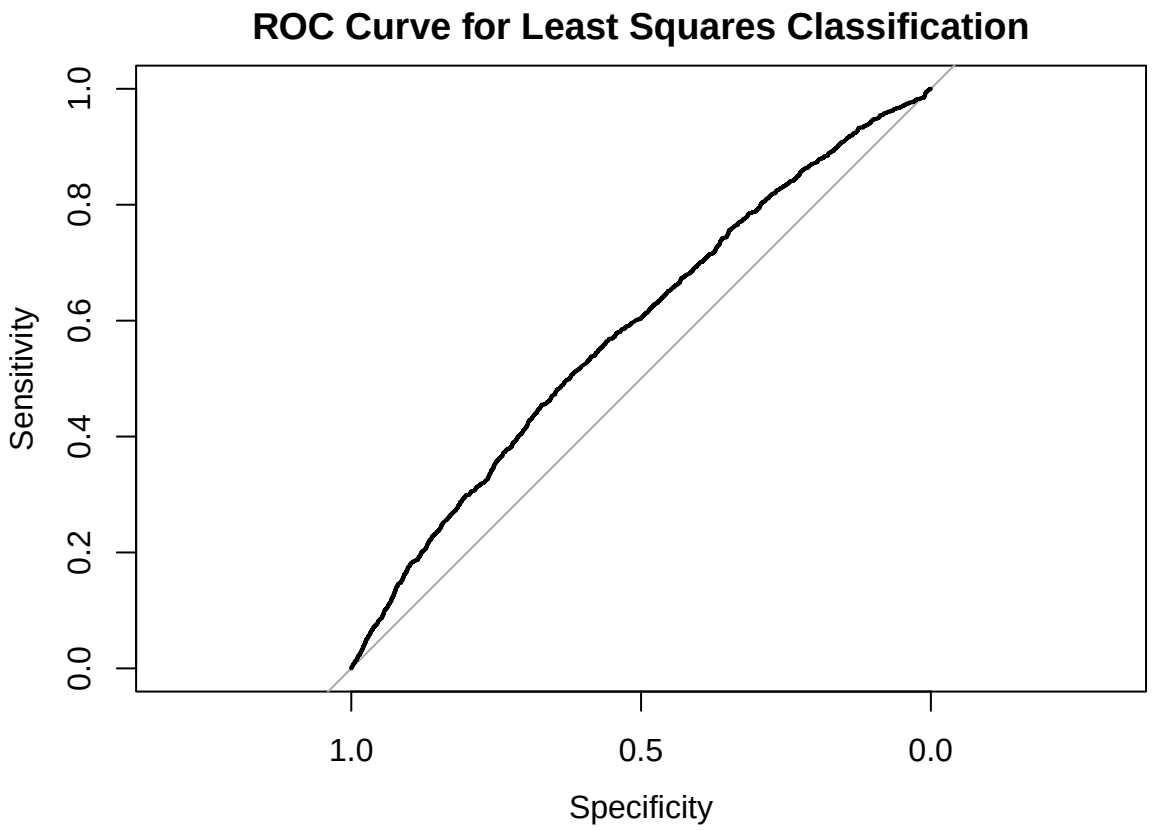
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     0     1
##          0 11976  1584
##          1     0     2
##
##                Accuracy : 0.8832
##                  95% CI : (0.8777, 0.8886)
##     No Information Rate : 0.8831
##     P-Value [Acc > NIR] : 0.4854
##
##                   Kappa : 0.0022
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 1.000000
##             Specificity : 0.001261
##          Pos Pred Value : 0.883186
##          Neg Pred Value : 1.000000
##              Prevalence : 0.883056
##          Detection Rate : 0.883056
##    Detection Prevalence : 0.999853
##       Balanced Accuracy : 0.500631
##
##        'Positive' Class : 0
##
```

```
# Plot an ROC curve for the test predictions
roc_obj <- roc(test_data$y_num, y_hat_test)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
plot(roc_obj, main = "ROC Curve for Least Squares Classification")
```




```
auc_val <- auc(roc_obj)
print(auc_val)
```

```
## Area under the curve: 0.5832
```

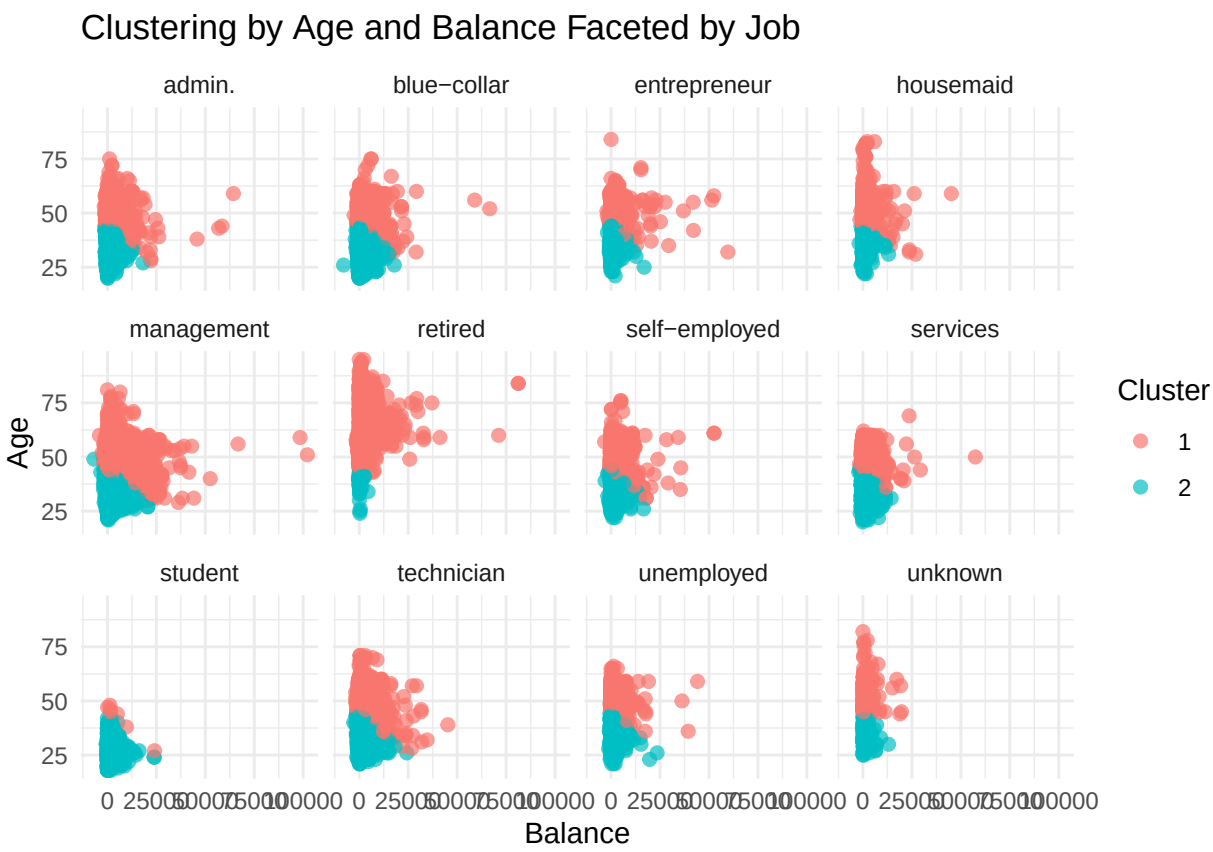
```
# Read the dataset and select key variables (you can include age if needed)
bank <- read.csv("bank-full.csv", sep = ";", header = TRUE)
# Filter for the "management" job (change to the desired job category)
bank_management <- bank %>%
  filter(job == "management") %>%
  select(marital, education, age, balance, y)

bank_management <- bank_management[sample(nrow(bank_management), 100), ]

# Convert categorical variables to numeric
bank_management$marital <- as.numeric(factor(bank_management$marital))
bank_management$education <- as.numeric(factor(bank_management$education))
bank_management$age <- as.numeric(factor(bank_management$age))
bank_management$y <- as.numeric(factor(bank_management$y))  # Assuming y is the loan status

# Scale the data to ensure all features contribute equally
bank_management_scaled <- scale(bank_management)


# Perform k-means clustering with k = 2 (you can change centers as needed)
set.seed(123)
km_mgmt <- kmeans(bank_management_scaled, centers = 2, nstart = 20)
```

```r
# Add the cluster assignments to the subset
bank_management$cluster <- factor(km_mgmt$cluster)

# Visualize the clustering results using a scatter plot
ggplot(bank_management, aes(x = balance, y = age, color = cluster)) +
  geom_point(size = 3, alpha = 0.8) +
  labs(title = "K-means Clustering for Management Job",
       x = "Balance",
       y = "age",
       color = "Cluster") +
  theme_minimal()
```


```r
library(dplyr)
bank_management %>%
  group_by(cluster) %>%
  summarize(
    avg_age       = mean(age),
    avg_balance   = mean(balance),
    avg_education = mean(education),  # if you included it in the model
    prop_loan     = mean(y)           # if y is numeric 0/1
  )
```

```
## # A tibble: 2 x 5
##   cluster avg_age avg_balance avg_education prop_loan
##   <fct>     <dbl>       <dbl>         <dbl>     <dbl>
## 1 1          9.37       1658.          2.93      1.30
## 2 2         22.1        2833.          2.85      1.17
```

```r
library(dplyr)
library(ggplot2)

# Read the dataset and select key variables
bank <- read.csv("bank-full.csv", sep = ";", header = TRUE)
bank_subset <- bank %>%
  select(job, marital, education, age, balance, y)

# Convert necessary columns
# We'll convert job to a factor for faceting, and also make sure age and balance are numeric.
bank_subset$job <- as.factor(bank_subset$job)
bank_subset$age <- as.numeric(as.character(bank_subset$age))
bank_subset$balance <- as.numeric(as.character(bank_subset$balance))
# Recode loan status if needed; here we assume y is already meaningful or recode later

# Optional: if you want to include only a subset for performance, uncomment the next line:
#bank_subset <- bank_subset %>% sample_n(1000)

# If you have clustering results (say, from k-means) on the whole dataset, merge them back.
# For illustration, let's run k-means clustering on the scaled numeric variables:
# (We'll use age and balance for simplicity; you can include more variables as needed)
set.seed(123)
bank_cluster_data <- bank_subset %>%
  select(age, balance) %>%
  scale()

km_out <- kmeans(bank_cluster_data, centers = 2, nstart = 20)
bank_subset$cluster <- factor(km_out$cluster)

# Create a facet wrap plot by job category
ggplot(bank_subset, aes(x = balance, y = age, color = cluster)) +
  geom_point(alpha = 0.7, size = 2) +
  facet_wrap(~ job) +
  labs(title = "Clustering by Age and Balance Faceted by Job",
       x = "Balance",
       y = "Age",
       color = "Cluster") +
  theme_minimal()
```




#Linear Regression Model of Balance vs Age

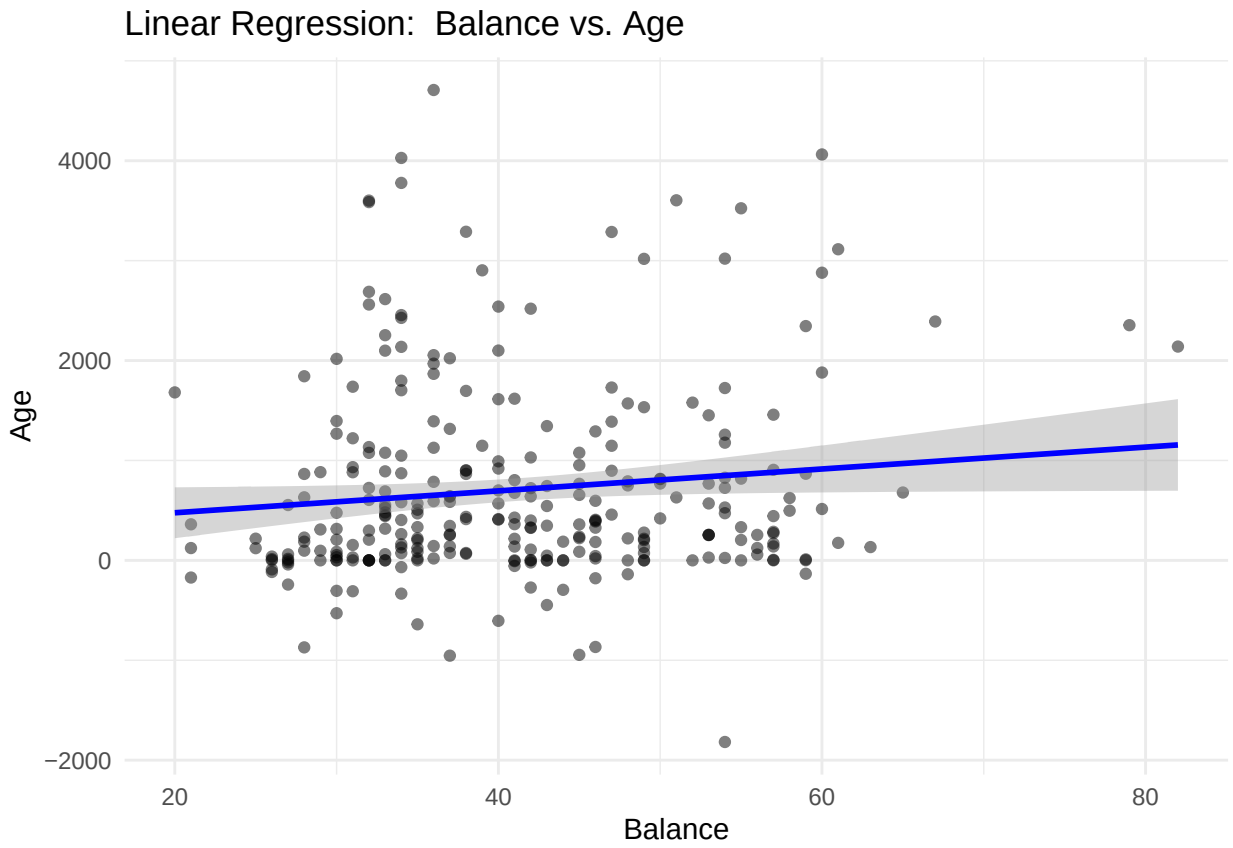
```
# Read in the dataset
bank <- read.csv("bank-full.csv", sep = ";", header = TRUE)

# Ensure that numeric variables are in numeric format and categorical variables are factors
bank$age <- as.numeric(bank$age)
bank$balance <- as.numeric(bank$balance)
bank$job <- as.factor(bank$job)
bank$marital <- as.factor(bank$marital)
bank$education <- as.factor(bank$education)
bank$y <- as.factor(bank$y)  # loan status

set.seed(2)
bank <- bank[sample(nrow(bank), 300), ]

lm_bank <- bank %>%
  filter(balance < 5000)

# Fit a linear regression model predicting 'balance' from age, job, marital, and education
model <- lm(balance ~ age + job + marital + education, data = lm_bank)

# View the summary of the model
summary(model)
```

```
##
## Call:
## lm(formula = balance ~ age + job + marital + education, data = lm_bank)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2399.1  -580.5  -250.2   288.1  3633.4
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)          -17.32     438.54  -0.039   0.9685
## age                   15.21       7.11   2.140   0.0333 *
## jobblue-collar       -28.76     222.11  -0.129   0.8971
## jobentrepreneur     -323.01     384.20  -0.841   0.4013
## jobhousemaid        -443.44     353.51  -1.254   0.2108
## jobmanagement       -551.49     244.45  -2.256   0.0249 *
## jobretired          -305.52     336.43  -0.908   0.3646
## jobself-employed    -313.58     345.47  -0.908   0.3649
## jobservices         -603.46     254.55  -2.371   0.0185 *
## jobstudent           347.92     718.60   0.484   0.6287
## jobtechnician       -392.61     230.66  -1.702   0.0899 .
## jobunemployed        -14.32     351.33  -0.041   0.9675
## maritalmarried       169.57     171.76   0.987   0.3244
## maritalsingle         64.82     209.80   0.309   0.7576
## educationsecondary   194.73     184.31   1.057   0.2917
## educationtertiary    544.31     231.63   2.350   0.0195 *
## educationunknown     210.05     383.68   0.547   0.5845
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 964.2 on 267 degrees of freedom
## Multiple R-squared:  0.07541,    Adjusted R-squared:   0.02
## F-statistic: 1.361 on 16 and 267 DF,  p-value: 0.161
```

```
# Optional: Visualize the relationship between age and balance with a regression line
ggplot(lm_bank, aes(x = age, y = balance)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "lm", color = "blue") +
  labs(title = "Linear Regression:  Balance vs. Age",
       x = "Balance",
       y = "Age") +
  theme_minimal()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

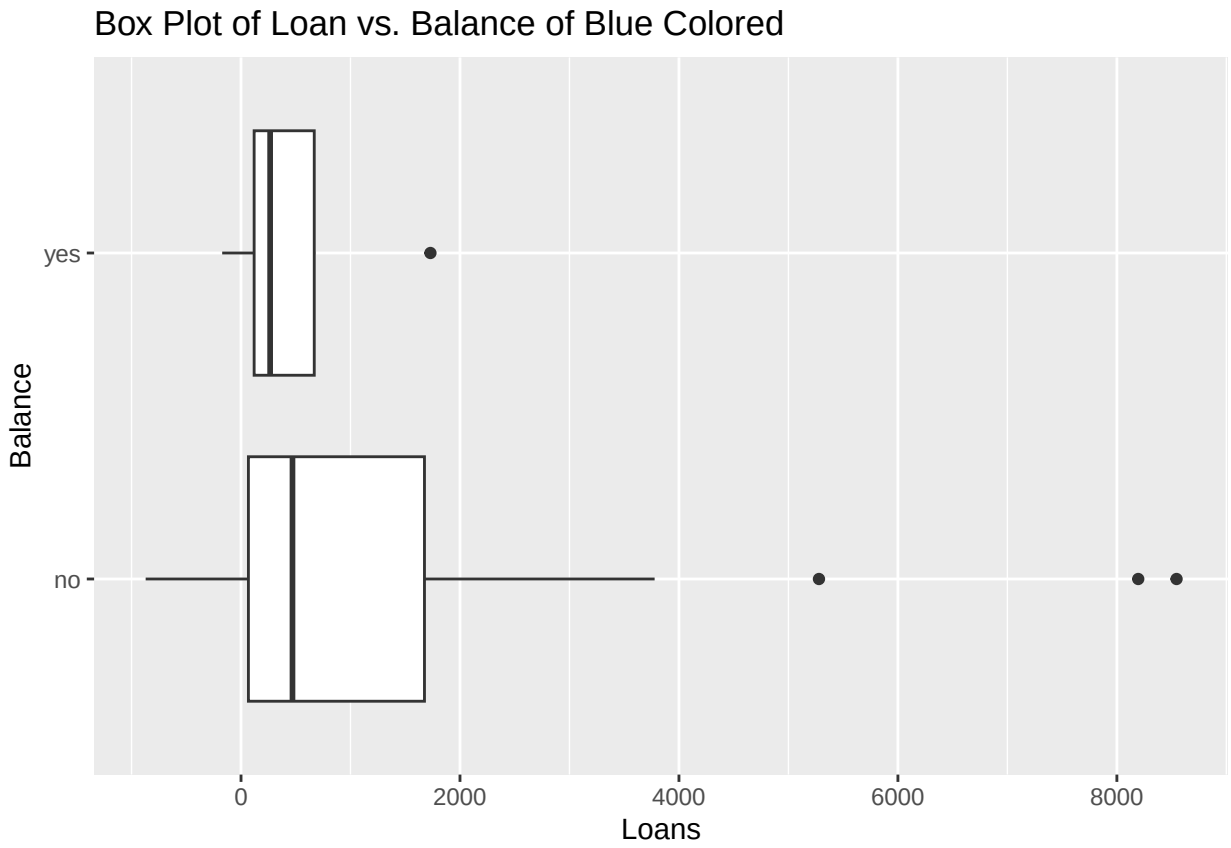



It shows that as the age goes up, you have more money in the bank account

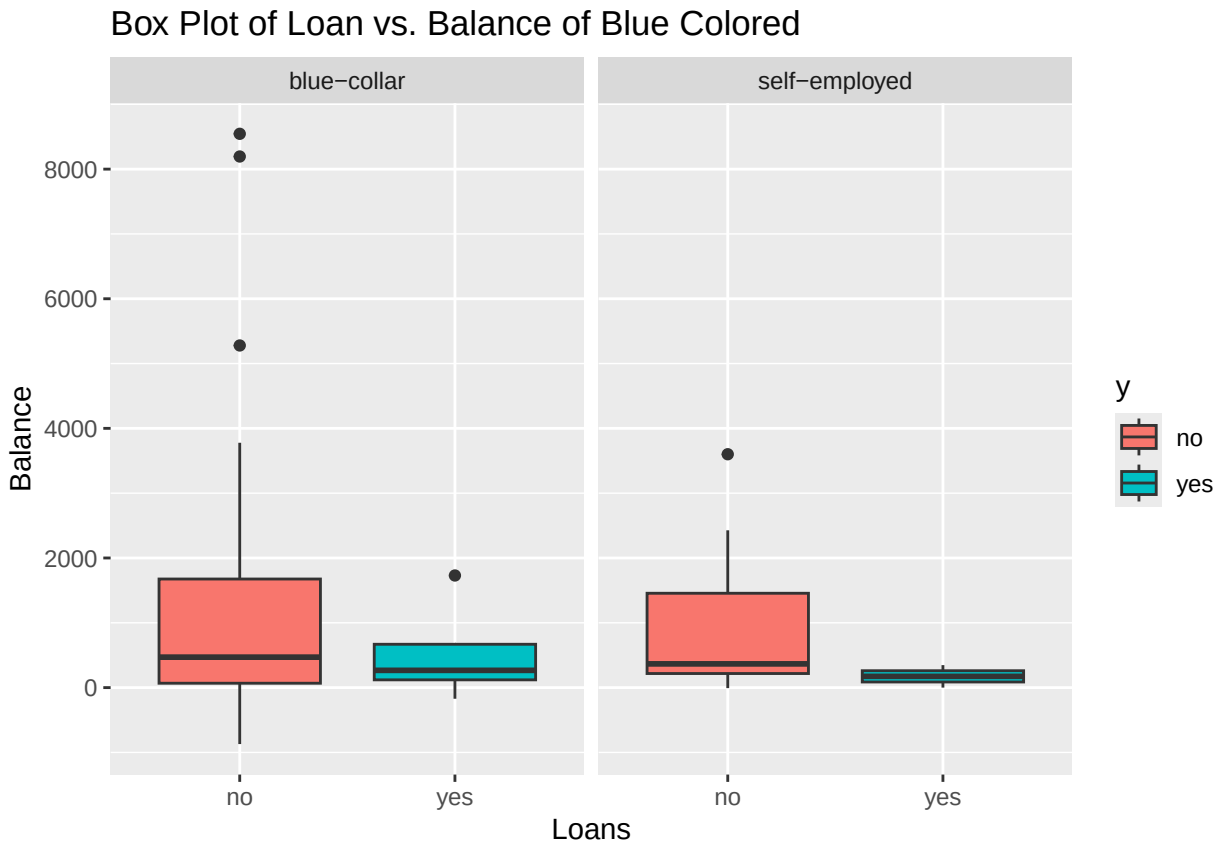
#Box Plot by jobs to see whether they have loans or not

```
bank_self_employed <- bank %>%
  filter(job == "self-employed")

bank_blue_color <- bank %>%
  filter(job == "blue-collar")

ggplot(bank_self_employed, aes(balance, y)) +
  geom_boxplot() +
  labs(title = "Box Plot of Loan vs. Balance of Self-Employed", x = "Loans", y = "Balance")
```

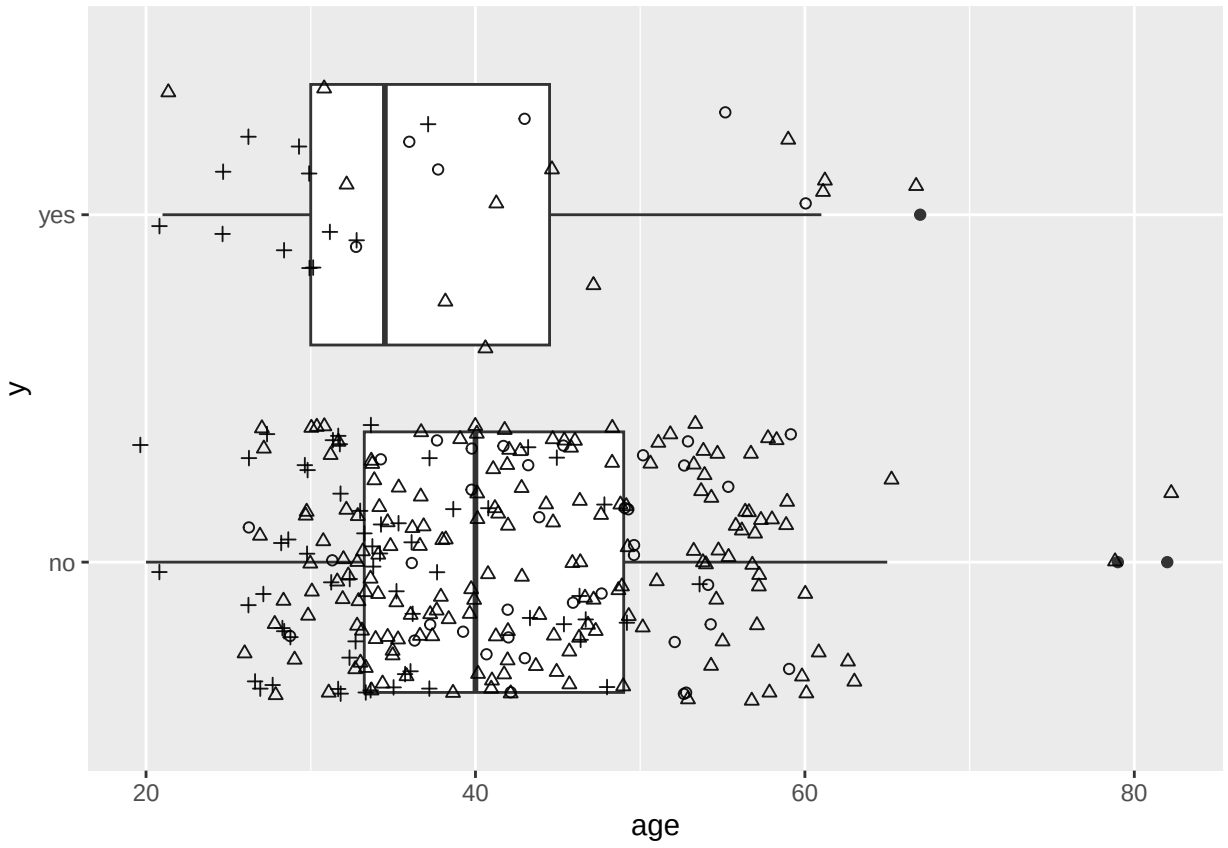

```
ggplot(bank_blue_color, aes(balance, y)) +
  geom_boxplot() +
  labs(title = "Box Plot of Loan vs. Balance of Blue Colored", x = "Loans", y = "Balance")
```



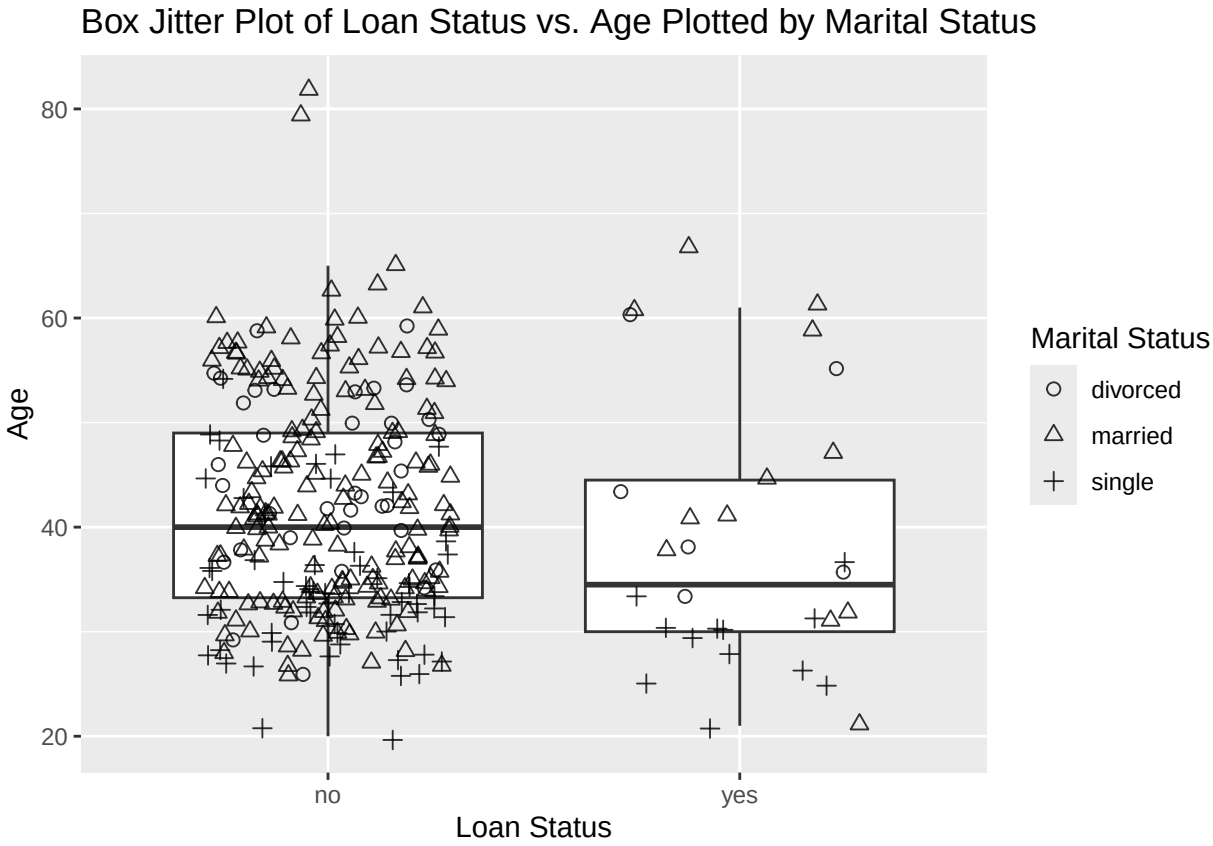

```
bank_blue_self <- bank %>%
  filter(job == "blue-collar" | job == "self-employed")

ggplot(bank_blue_self, aes(y, balance, fill=y)) +
  geom_boxplot() +
  facet_wrap(~job) +
  labs(title = "Box Plot of Loan vs. Balance of Blue Colored", x = "Loans", y = "Balance")
```




```
bank_self_employed <- bank %>%
  filter(job == "marital")

bank_blue_color <- bank %>%
  filter(job == "blue-collar")

ggplot(bank, aes(age, y)) +
  geom_boxplot() +
  #+geom_point(shape = bank$marital)
  geom_jitter(shape = bank$marital, size=1.5, alpha=0.9) +
  labs(shape = "Marital Status")
```




```
ggplot(bank, aes(x = y, y = age)) +
  # One boxplot per loan status (yes/no)
  geom_boxplot(outlier.shape = NA) +
  geom_jitter(aes(shape = marital),
              width = 0.3,   # spread out points horizontally
              size = 2,
              alpha = 0.8) +
  # Legend label for shapes
  labs(title = "Box Jitter Plot of Loan Status vs. Age Plotted by Marital Status",
       shape = "Marital Status",
       x = "Loan Status",
       y = "Age")  +
  scale_shape_manual(values = c("divorced" = 1, "married" = 2, "single" = 3))
```




Single have higher percentage of having loan than married.

The question sounds like more classification rather than clustering.