QuantitativeFinanceProject

2023-09-24

Contents

1 Group assignment of introduction to Quantitative Finance

Hand-in date: 22.10.2023
Campus: BI Oslo
Examination code : ELE 3911
Examination name: Introduction to Quantitative Finance

1.1 Introduction

Within this appendix, we provide a detailed overview of the R code used in the quantitative analysis of the data-set in the context of the Introduction to Quantitative Finance coursework assignment. The R code is accompanied by explanatory comments written in Markdown. To help the reader, the code sections are enumerated to facilitate cross-referencing with the main text. This supplementary material is designed to assist in comprehending the quantitative methods applied and the solutions provided in the main solution paper.

1.2 Prepare

In the code sections 1-2 we will load the libraries useful to our analysis, import the data-set and clean the data.

[code section: 1]

```r
# IMPORTING PACKAGES

# install.packages('formatR') install.packages('ggplot2')
# install.packages('moments')

library(ggplot2)
library(moments)

set.seed(42)
```

[code section: 2]

```r
# IMPORTING AND CLEANING P2P LOANS DATASET

# importing data-set as data-frame from .csv file with relative path
p2ploans = read.csv("p2ploans.csv", sep = ",", dec = ".", header = T, colClasses = "character")

print("P2P loans dataframe:")
```

```
## [1] "P2P loans dataframe:"
```

```r
str(p2ploans)
```

```
## 'data.frame':    5000 obs. of  7 variables:
##  $ id             : chr  "1" "2" "3" "4" ...
##  $ interest_rate  : chr  "13.8" "30.92" "20.66" "14.05" ...
##  $ internal_rating: chr  "C" "HR" "D" "C" ...
##  $ maturity       : chr  "5" "3" "5" "5" ...
##  $ dti_ratio      : chr  "16" "49" "31" "30" ...
##  $ risk_free      : chr  "1.72" "1.72" "1.72" "1.72" ...
##  $ yearly_payment : chr  "3410.64" "1044.24" "4936.68" "6244.2" ...
```

```r
# converting numerical variables in numeric data type
for (tempVar in c("interest_rate", "maturity", "dti_ratio", "risk_free", "yearly_payment")) {
```

```
    p2ploans[[tempVar]] <- as.numeric(p2ploans[[tempVar]])
}
```

2 Task 1

2.1 1.a

The following chunk of code computes the mean and median statistics for the yearly payment variable of the data-set, exploiting the related built-in functions of R.

[code section: 3]

```
# COMPUTING STATISTICS OF THE yearly_payment VARIABLE

meanYearlyPayment = mean(p2ploans$yearly_payment)
medianYearlyPayment = median(p2ploans$yearly_payment)

cat("The mean of the yearly payments is:", meanYearlyPayment, "\n")
```

```
## The mean of the yearly payments is: 4918.378
```

```
cat("The median of the yearly payments is:", medianYearlyPayment)
```

```
## The median of the yearly payments is: 4322.94
```

We used the “summary” function to gain a deeper understanding of our data-set, its useful to observe some more statistics about the distribution of our variables.

[code section: 4]

```
# COMPUTING STATISTICS OF THE VARIABLES

summary(p2ploans)
```

```
##       id             interest_rate   internal_rating       maturity
##  Length:5000        Min.   : 4.32   Length:5000        Min.   :3.000
##  Class :character   1st Qu.: 8.70   Class :character   1st Qu.:3.000
##  Mode  :character   Median :13.80   Mode  :character   Median :3.000
##                     Mean   :15.54                      Mean   :3.634
##                     3rd Qu.:22.05                      3rd Qu.:5.000
##                     Max.   :30.92                      Max.   :5.000
##    dti_ratio      risk_free     yearly_payment
##  Min.   : 0.0   Min.   :1.72   Min.   :  497.8
##  1st Qu.:18.0   1st Qu.:1.72   1st Qu.: 3086.2
##  Median :26.0   Median :1.72   Median : 4322.9
##  Mean   :27.6   Mean   :1.72   Mean   : 4918.4
##  3rd Qu.:35.0   3rd Qu.:1.72   3rd Qu.: 6294.7
##  Max.   :68.0   Max.   :1.72   Max.   :15020.2
```

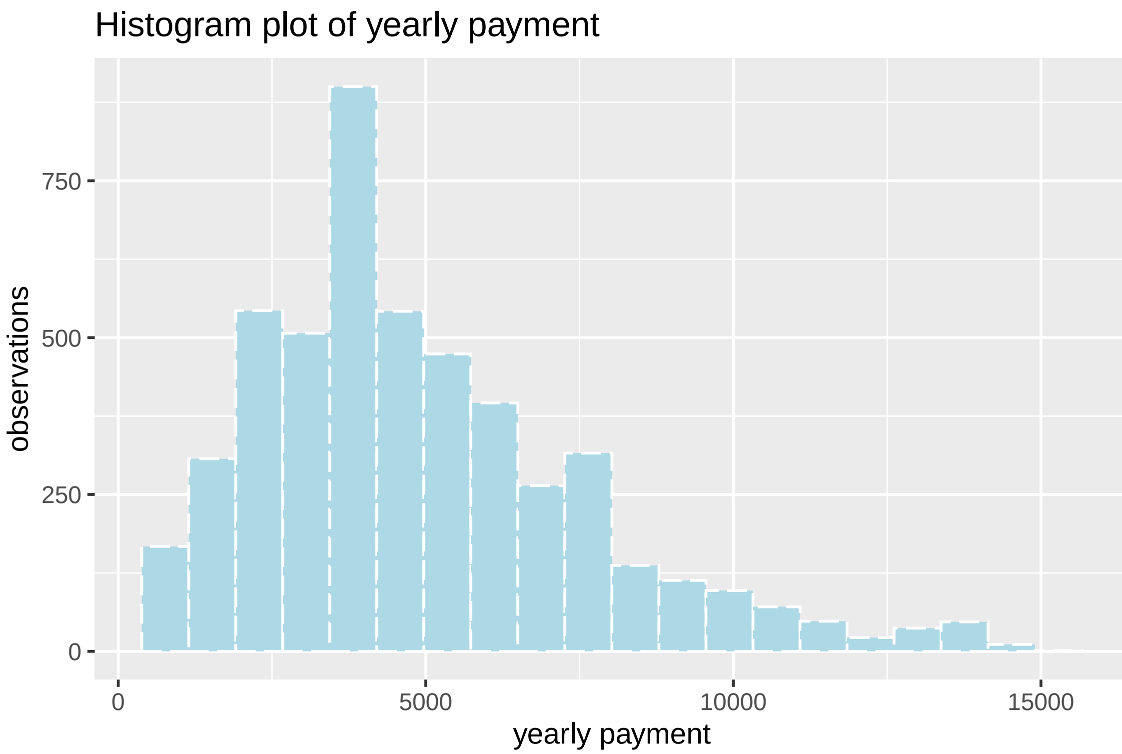
2.2 1.b

In the following lines we used ggplot to construct a histogram of the yearly payment with 20 bins.

[code section: 5]

```
# HISTOGRAM PLOT OF THE YEARLY PAYMENT

ggplot(p2ploans, aes(x = yearly_payment)) + geom_histogram(colour = "white", fill = "lightblue",
    bins = 20, linetype = "longdash") + ggtitle("Histogram plot of yearly payment") +
    xlab("yearly payment") + ylab("observations")
```



2.3 1.d

The following chunk of code computes the skewness and kurtosis statistics for the yearly payment variable of the data-set, exploiting the related built-in functions of R.

[code section: 6]

```
skewnessYearlyPayment = skewness(p2ploans$yearly_payment)
kurtosisYearlyPayment = kurtosis(p2ploans$yearly_payment)

cat("The skeweness of the yearly payments is:", skewnessYearlyPayment, "\n")
```

```
## The skeweness of the yearly payments is: 1.019249
```

```
cat("The kurtosis of the yearly payments is:", kurtosisYearlyPayment)
```

```
## The kurtosis of the yearly payments is: 4.073969
```

2.4 1.e

In the chunk that follows we set the parameters for the triangular distribution for modelling yearly payment based on p2ploans data.

[code section: 7]

```
# TRIANGULAR DISTRIBUTION TO MODEL YEARLY PAYMENTS

# computing the approximate mode, useful for continuous data like yearly
# payment
custom_mode <- function(x) {
    table_x <- table(x)
    mode_value <- as.numeric(names(table_x[table_x == max(table_x)]))
    return(mode_value)
```

```
}

minYearlyPayment = min(p2ploans$yearly_payment)
maxYearlyPayment = max(p2ploans$yearly_payment)
modeYearlyPayment = custom_mode(round(p2ploans$yearly_payment, digits = -2))

cat("The min value of the yearly payments is:", minYearlyPayment, "\n")
```

```
## The min value of the yearly payments is: 497.76
```

```
cat("The max value of the yearly payments is:", maxYearlyPayment, "\n")
```

```
## The max value of the yearly payments is: 15020.16
```

```
cat("The mode of the yearly payments is:", modeYearlyPayment)
```

```
## The mode of the yearly payments is: 3900
```

2.5 1.f

As depicted in the code excerpt below, we identified the loan with the largest yearly payment and, assuming that payments are made at the end of each year and that we discount payments at the risk free rate of 1.72%, we computed the present value of the first yearly payment made on the loan.

[code section: 8]

```
# FIRST YEAR PRESENT VALUE COMPUTATIONS ON HIGHEST YEARLY PAYMENT LOAN

# finding loan with the highest yearly payment and reporting its id
largestYearlyPaymentLoan = subset(p2ploans, yearly_payment == max(p2ploans$yearly_payment))
largestYearlyPaymentLoan_id = largestYearlyPaymentLoan$id
cat("ID associated to the loan with the highest yearly payment:", largestYearlyPaymentLoan_id,
    "\n")
```

```
## ID associated to the loan with the highest yearly payment: 137
```

```
# computing the present value of the first yearly payment of the loan
largestYearlyPayment_presentValue_year1 = largestYearlyPaymentLoan$yearly_payment *
    (1 + largestYearlyPaymentLoan$risk_free/100)^(-1)
cat("Present value of the first yearly payment made on the loan with the highest yearly payment:",
    largestYearlyPayment_presentValue_year1)
```

```
## Present value of the first yearly payment made on the loan with the highest yearly payment: 14766.18
```

2.6 1.g

Below, we computed the present value of yearly payments to the platform over the duration of the loan.

[code section: 9]

```
# PRESENT VALUE COMPUTATIONS ON HIGHEST YEARLY PAYMENT LOAN

largestYearlyPayment_presentValue_year2 = largestYearlyPaymentLoan$yearly_payment *
    (1 + largestYearlyPaymentLoan$risk_free/100)^(-2)
largestYearlyPayment_presentValue_year3 = largestYearlyPaymentLoan$yearly_payment *
    (1 + largestYearlyPaymentLoan$risk_free/100)^(-3)
largestYearlyPayment_presentValue_year4 = largestYearlyPaymentLoan$yearly_payment *
    (1 + largestYearlyPaymentLoan$risk_free/100)^(-4)
largestYearlyPayment_presentValue_year5 = largestYearlyPaymentLoan$yearly_payment *
```

```r
    (1 + largestYearlyPaymentLoan$risk_free/100)^(-5)

LargestYearlyPayment_presentValue = sum(largestYearlyPayment_presentValue_year1,
    largestYearlyPayment_presentValue_year2, largestYearlyPayment_presentValue_year3,
    largestYearlyPayment_presentValue_year4, largestYearlyPayment_presentValue_year5)

cat("Present value of the payments made on the loan with the highest yearly payment:",
    LargestYearlyPayment_presentValue)
```

```
## Present value of the payments made on the loan with the highest yearly payment: 71375.93
```

3 Task 2

3.1 2.a

Moving forward in the code, we computed the expected value of of the first yearly payment for the loan with id = 5, assuming probability of default of 0.05 for loans that have not entered default, irrespective of the year of payment.

[code section: 10]

```
# EXPECTED VALUE OF FIRST YEARLY PAYMENT OF LOAN WITH ID = 5

# taking data of the loan
id5Loan = subset(p2ploans, id == 5)
cat("The maturity of the loan with id = 5 is:", id5Loan$maturity, "years \n")
```

```
## The maturity of the loan with id = 5 is: 3 years
```

```
# defining probability of default at first year and computing probability of
# non default
pd_1year = 0.05  # probability of default at first year
pnd_1year = 1 - pd_1year  # probability of non default at first year

# computing expected value
id5Loan_expectedValue_year1 = id5Loan$yearly_payment * pnd_1year
cat("The expected value of first yearly payment of loan with id = 5 is", id5Loan_expectedValue_year1)
```

```
## The expected value of first yearly payment of loan with id = 5 is 1967.184
```

3.2 2.b

Below, we computed the expected value of the final yearly payment of the loan with id = 5.

[code section: 11]

```
# EXPECTED VALUE OF FINAL YEARLY PAYMENT OF LOAN WITH ID = 5

# computing probability of default and non default at third year
pd_3year = pd_1year + pnd_1year * pd_1year + pnd_1year * pnd_1year * pd_1year  # probability of default
pnd_3year = pnd_1year^3  # probability of non default at last (third) year

# checking that probabilities sum to 1
flag = pd_3year + pnd_3year
cat("The sum of probabilities of default and non default at third year is", flag,
    "\n")
```

```
## The sum of probabilities of default and non default at third year is 1
```

```
id5Loan_expectedValue_year3 = id5Loan$yearly_payment * pnd_3year
cat("The expected value of final (third) yearly payment of loan with id = 5 is",
    id5Loan_expectedValue_year3)
```

```
## The expected value of final (third) yearly payment of loan with id = 5 is 1775.384
```

3.3 2.e

In the course of the following code, we computed the expected number of defaults in the first year, the variance of the number of defaults and the skewness of the number of defaults.

[code section: 12]

```
# EXPECTED VALUE, VARIANCE AND SKEWENESS OF DEFAULTS IN THE FIRST YEAR
# (CONSIDERING ALL LOANS)

defaults_expectedValue_1year = pd_1year * nrow(p2ploans)  # expected value
defaults_variance_1year = nrow(p2ploans) * pd_1year * (1 - pd_1year)  # variance
defaults_skeweness_1year = ((1 - pd_1year) - pd_1year)/sqrt(defaults_variance_1year)  # skewness

cat("The expected number of defaults in the first year is", defaults_expectedValue_1year,
    "\n")
```

```
## The expected number of defaults in the first year is 250
```

```
cat("The varince of the number of defaults in the first year is", defaults_variance_1year,
    "\n")
```

```
## The varince of the number of defaults in the first year is 237.5
```

```
cat("The skeweness of the number of defaults in the first year is", defaults_skeweness_1year)
```

```
## The skeweness of the number of defaults in the first year is 0.05839971
```

```
# expected because it's five percent of the total
```

3.4 2.f

In the chunk that comes next, we considered the aggregate yearly payment flows to the platform from the loans with id values of 1-10, and computed the sum of expected payments at the end of the first year, assuming that the default probability for each loan is 0.05.

[code section: 13]

```
# SUM OF EXPECTED PAYMENTS OF LOANS WITH IDs 1-10 AT THE END OF FIRST YEAR

expectedReturns <- (1 - pd_1year) * p2ploans[p2ploans$id %in% 1:10, ]$yearly_payment
sumExpectedReturs <- sum(expectedReturns)
cat("The sum of the expected payments of loans with ids 1-10 at the end of the first year is",
    sumExpectedReturs)
```

```
## The sum of the expected payments of loans with ids 1-10 at the end of the first year is 41669.51
```

4 Task 3

4.1 3.a

In the following lines of code we computed the mean of the interest rate for AA and HR groups, and their difference in percentage points.

[code section: 14]

```
# MEAN OF THE INTEREST RATE FOR AA AND HR GROUPS

meanInterestRate_AAgroup <- mean(p2ploans[p2ploans$internal_rating == "AA", ]$interest_rate)
meanInterestRate_HRgroup <- mean(p2ploans[p2ploans$internal_rating == "HR", ]$interest_rate)
percentagePointsDifference <- abs((meanInterestRate_AAgroup - meanInterestRate_HRgroup))

cat("The mean interest rate for the AA group is", meanInterestRate_AAgroup, "\n")
```

```
## The mean interest rate for the AA group is 5.647083
```

```
cat("The mean interest rate for the HR group is", meanInterestRate_HRgroup, "\n")
```

```
## The mean interest rate for the HR group is 30.34732
```

```
cat("Difference in Percentage Points:", percentagePointsDifference)
```

```
## Difference in Percentage Points: 24.70023
```

```
# AA is less risky than HR
```

4.2 3.c

Within the chunks below, we regressed the interest rate on the internal rating, exploiting the lm() built-in function of R. While in code section 15 the model takes into account the intercept and discard the factor(internal_rating)A, in code section 16 a new model is implemented, which takes into account all the factors and discards the intercept.

[code section: 15]

```
# LINEAR REGRESSION MODEL FOR INTEREST RATE ON THE INTERNAL RATINGS

model <- lm(interest_rate ~ factor(internal_rating), data = p2ploans)
summary(model)
```

```
##
## Call:
## lm(formula = interest_rate ~ factor(internal_rating), data = p2ploans)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.3143 -0.9471  0.0805  0.7605  3.5842
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 7.97951    0.04755  167.81   <2e-16 ***
## factor(internal_rating)AA -2.33243    0.07612  -30.64   <2e-16 ***
## factor(internal_rating)B   3.01974    0.06878   43.90   <2e-16 ***
## factor(internal_rating)C   7.47501    0.06601  113.25   <2e-16 ***
## factor(internal_rating)D  13.42630    0.06842  196.24   <2e-16 ***
## factor(internal_rating)E  18.77475    0.07993  234.89   <2e-16 ***
```

```
## factor(internal_rating)HR 22.36781    0.08610  259.79   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.427 on 4993 degrees of freedom
## Multiple R-squared:  0.9675, Adjusted R-squared:  0.9674
## F-statistic: 2.476e+04 on 6 and 4993 DF,  p-value: < 2.2e-16
```

[code section: 16]

```
# LINEAR REGRESSION MODEL FOR INTEREST RATE ON THE INTERNAL RATINGS NOT
# CONSIDERING THE INTERCEPT

model_nointercep <- lm(interest_rate ~ 0 + factor(internal_rating), data = p2ploans)
summary(model)
```

```
##
## Call:
## lm(formula = interest_rate ~ factor(internal_rating), data = p2ploans)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.3143 -0.9471  0.0805  0.7605  3.5842
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 7.97951    0.04755  167.81   <2e-16 ***
## factor(internal_rating)AA -2.33243    0.07612  -30.64   <2e-16 ***
## factor(internal_rating)B   3.01974    0.06878   43.90   <2e-16 ***
## factor(internal_rating)C   7.47501    0.06601  113.25   <2e-16 ***
## factor(internal_rating)D  13.42630    0.06842  196.24   <2e-16 ***
## factor(internal_rating)E  18.77475    0.07993  234.89   <2e-16 ***
## factor(internal_rating)HR 22.36781    0.08610  259.79   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.427 on 4993 degrees of freedom
## Multiple R-squared:  0.9675, Adjusted R-squared:  0.9674
## F-statistic: 2.476e+04 on 6 and 4993 DF,  p-value: < 2.2e-16
```

4.3 3.d

Within the following lines of code, we added to the model created in code section 16 two new regressors: dti_ratio and maturity.

[code section: 17]

```
# LINEAR REGRESSION MODEL FOR INTEREST RATE USING INTERNAL RATINGS, DTI RATIO
# AND MATURITY

model_dtiMaturity <- lm(interest_rate ~ 0 + factor(internal_rating) + dti_ratio +
    maturity, data = p2ploans)
summary(model_dtiMaturity)
```

```
##
## Call:
## lm(formula = interest_rate ~ 0 + factor(internal_rating) + dti_ratio +
```

```
##     maturity, data = p2ploans)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.5473 -0.9607  0.0705  0.7962  3.8545
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## factor(internal_rating)A   7.643022   0.100149  76.316  < 2e-16 ***
## factor(internal_rating)AA  5.332922   0.104428  51.068  < 2e-16 ***
## factor(internal_rating)B  10.645324   0.105139 101.250  < 2e-16 ***
## factor(internal_rating)C  15.076857   0.105648 142.708  < 2e-16 ***
## factor(internal_rating)D  21.005954   0.108332 193.903  < 2e-16 ***
## factor(internal_rating)E  26.342827   0.116668 225.794  < 2e-16 ***
## factor(internal_rating)HR 29.941247   0.119128 251.337  < 2e-16 ***
## dti_ratio                  0.009817   0.001679   5.848 5.29e-09 ***
## maturity                   0.026804   0.021721   1.234    0.217
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.422 on 4991 degrees of freedom
## Multiple R-squared:  0.9934, Adjusted R-squared:  0.9933
## F-statistic: 8.296e+04 on 9 and 4991 DF,  p-value: < 2.2e-16
```

5 Task 4

5.1 4.a

In the subsequent lines of code, we simulated the first year of payments 1000 times for all loans in the E and HR internal rating groups (description of the algorithm deployed is included in the solution paper): we used this data to construct a histogram with 20 bins.

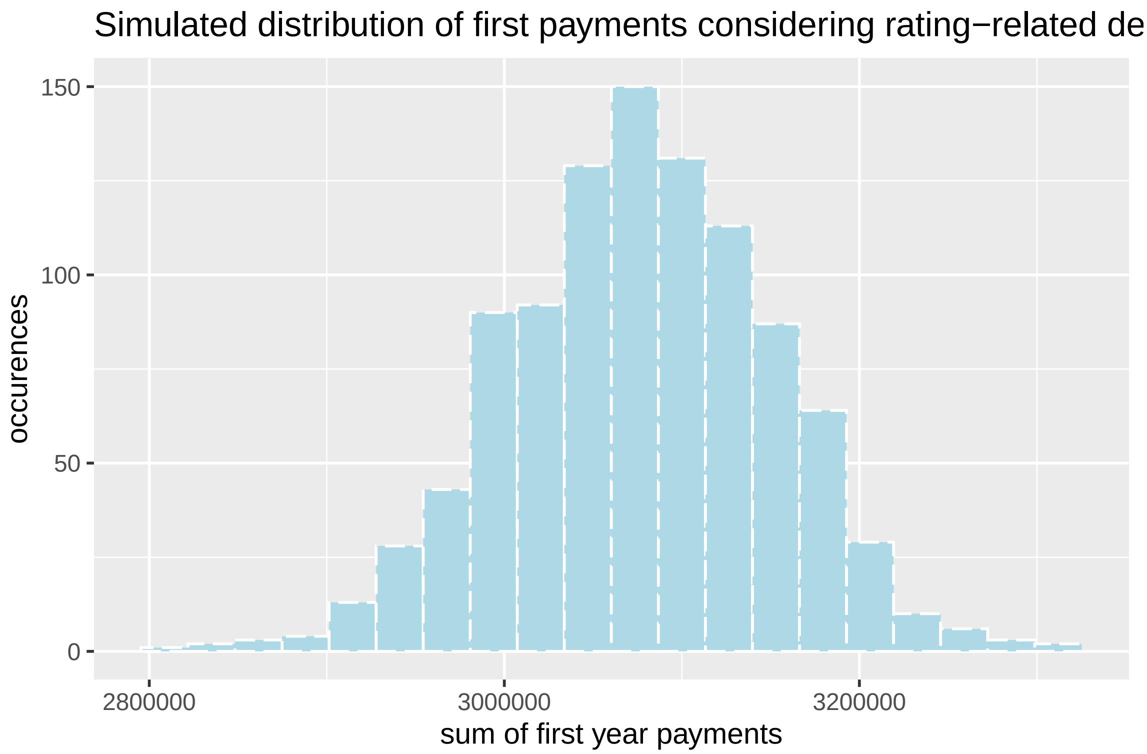
[code section: 18]

```
# SIMULATING FOR 1000 TIMES THE SUCCESS OF THE FIRST YEAR PAYMENTS OF LOANS IN
# THE E AND HR INTERNAL RATING GROUPS
loansEHR = p2ploans[p2ploans$internal_rating %in% c("E", "HR"), ]

sumOfPayments_df = data.frame(simulation_number = numeric(0), sum_of_payments = numeric(0))

for (i in 1:1000) {
    value = 0
    for (j in 1:nrow(loansEHR)) {
        trial = runif(1)
        if (p2ploans[j, ]$internal_rating == "E") {
            if (trial >= 0.15) {
                value = value + p2ploans[j, ]$yearly_payment
            }
        } else {
            if (trial >= 0.3) {
                value = value + p2ploans[j, ]$yearly_payment
            }
        }
    }
    new_row = data.frame(simulation_number = i, sum_of_payments = value)
    sumOfPayments_df <- rbind(sumOfPayments_df, new_row)
}

# histogram plot
ggplot(data = sumOfPayments_df, aes(x = sum_of_payments)) + geom_histogram(colour = "white",
    fill = "lightblue", bins = 20, linetype = "longdash") + ggtitle("Simulated distribution of first pa
    xlab("sum of first year payments") + ylab("occurences")
```



5.2 4.c

The following chunk of code computes the mean and standard deviation statistics for the yearly payment variable of the data-set, exploiting the related built-in functions of R. In code section 20, we computed also the expected value.

[code section: 19]

```
# MEAN AND MEDIAN VALUES OF SUM OF PAYMENTS OVER THE 1000 SIMULATIONS
meanSumOfPayments = mean(sumOfPayments_df$sum_of_payments)
standardDeviationSumOfPayments = sd(sumOfPayments_df$sum_of_payments)

cat("The mean of the sum of payments over the 1000 simulations is", meanSumOfPayments,
    "\n")
```

```
## The mean of the sum of payments over the 1000 simulations is 3076123
```

```
cat("The standard deviation of the sum of payments over the 1000 simulations is",
    standardDeviationSumOfPayments)
```

```
## The standard deviation of the sum of payments over the 1000 simulations is 74304.53
```

[code section: 20]

```
# EXPECTED VALUE OF SUM OF PAYMENTS OVER THE 1000 SIMULATIONS
expectedValueSumOfPayments = 0

for (i in 1:nrow(loansEHR)) {
    if (loansEHR[i, ]$internal_rating == "E") {
        expectedValueSumOfPayments = expectedValueSumOfPayments + p2ploans[i, ]$yearly_payment *
            0.85
    } else if (loansEHR[i, ]$internal_rating == "HR") {
        expectedValueSumOfPayments = expectedValueSumOfPayments + p2ploans[i, ]$yearly_payment *
```

```
            0.7
    }
}

cat("The expected value of the sum of payments over the 1000 simulations is", expectedValueSumOfPayments
```

```
## The expected value of the sum of payments over the 1000 simulations is 3369243
```

5.3 4.d

Continuing with the code, we computed the 95% VaR for total payments across the 1000 simulations.

[code section: 21]

```
# COMPUTATION OF VALUE AT RISK

VaR_95 = quantile(sumOfPayments_df$sum_of_payments, probs = 0.05)
cat("The 95% Value at Risk (VaR) is", VaR_95)
```

```
## The 95% Value at Risk (VaR) is 2953912
```