This is the main submission document. ***Save and rename this document filename with your <u>registered full name</u> as Prefix before submission.***

Class / Seminar Grp	4
Full Name	Nur Aisyah Bte Abdul Mutalib
Matriculation Number	U2110399H

** : Delete and replace as appropriate.*

Declaration of Academic Integrity

By submitting this assignment for assessment, I declare that this submission is my own work, unless otherwise quoted, cited, referenced or credited. I have read and understood the Instructions to CBA.PDF provided and the Academic Integrity Policy.

I am aware that failure to act in accordance with the University's Academic Integrity Policy may lead to the imposition of penalties which may include the requirement to revise and resubmit an assignment, receiving a lower grade, or receiving an F grade for the assignment; suspension from the University or termination of my candidature.

I consent to the University copying and distributing any or all of my work in any form and using third parties to verify whether my work contains plagiarised material, and for quality assurance purposes.

Please insert an "X" within the square brackets below to indicate your selection.

[X] I have read and accept the above.

Table of Contents

For each question, please start your answer in a new page.

Answer to Q1:

(a) It is necessary to factor Loan_ID, Gender, Married, Dependents, Education, Self_Employed, Loan_Amount_Term, Credit_Score, Property_Area and Loan_Status. This is because they are categorical variables with data that fall into discrete groups. Hence, storing this data as factors ensures that the modelling functions will treat such data correctly.

```
> sapply(homeloan2, class)
          Loan_ID            Gender           Married        Dependents 
      "character"       "character"       "character"       "character" 
        Education     Self_Employed   ApplicantIncome CoapplicantIncome 
      "character"       "character"         "integer"         "numeric" 
       LoanAmount  Loan_Amount_Term      Credit_Score     Property_Area 
        "integer"         "integer"         "integer"       "character" 
      Loan_Status 
      "character" 
```

Figure 1: List of datatypes of each variable before factoring

```
factors <- c("Loan_ID", "Gender", "Married", "Dependents", "Education",
"Self_Employed", "Loan_Amount_Term", "Credit_Score", "Property_Area",
"Loan_Status")
homeloan2[, (factors):= lapply(.SD, factor), .SDcols = factors]
```

Figure 2: Factoring the necessary variables

```
> sapply(homeloan2, class)
          Loan_ID            Gender           Married        Dependents 
         "factor"          "factor"          "factor"          "factor" 
        Education     Self_Employed   ApplicantIncome CoapplicantIncome 
         "factor"          "factor"         "integer"         "numeric" 
       LoanAmount  Loan_Amount_Term      Credit_Score     Property_Area 
        "integer"          "factor"          "factor"          "factor" 
      Loan_Status 
         "factor" 
```

Figure 3: List of datatypes of each variable after factoring

(b)

```
> colSums(is.na(homeloan2))
          Loan_ID            Gender           Married        Dependents         Education     Self_Employed 
                0                13                 2                13                 0                31 
  ApplicantIncome CoapplicantIncome        LoanAmount  Loan_Amount_Term      Credit_Score     Property_Area 
                0                 0                 0                14                49                 0 
      Loan_Status 
                0 
```

Figure 4: List of missing values for each variable

```
> nrow(homeloan2) # Total no. of rows
[1] 592
> sum(apply(homeloan2, 1, anyNA)) # No. of rows containing NA
[1] 112
> round((sum(apply(homeloan2, 1, anyNA))/nrow(homeloan2))*100,2) # % missing values
[1] 18.92
```

Figure 5: Percentage of missing values

The percentage of missing values is 18.92%. Therefore, we cannot ignore and drop the missing values as it will result in an 18.92% loss of data, more than the accepted percentage of 5%.

Therefore, I would impute the missing data using missForest, an implementation of the random forest algorithm. The missing data is imputed by building a random forest model for each variable which predicts missing values in the variable with the help of observed values[1].

```
install.packages("missForest")
library(missForest)

# Seed 10% missing values
homeloan2.mis <- prodNA(homeloan2[,-1], noNA = 0.1)

# Impute missing values, using all parameters as default values
homeloan2.imp <- missForest(homeloan2.mis)

# Check imputed values
homeloan2 <- homeloan2.imp$ximp

# Check imputation error
round(homeloan2.imp$OOBerror, 2)

# NRMSE   PFC
# 0.79  0.28
```

Figure 6: Using missForest to handle missing data

missForest is able to tell us that the dataset's continuous variables are imputed with 79% error and the dataset's categorical variables are imputed with 28% error.

[1] https://medium.com/coinmonks/dealing-with-missing-data-using-r-3ae428da2d17

Answer to Q2:

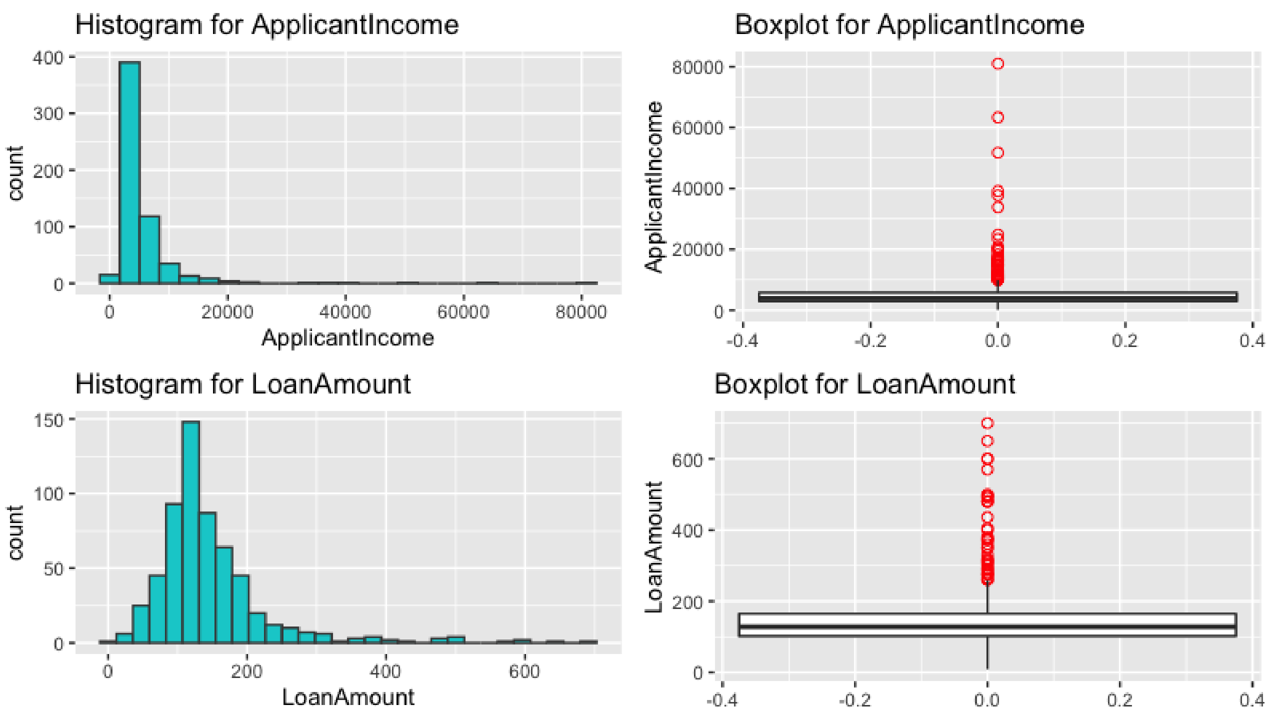

Figure 7: Histograms and Boxplots for ApplicantIncome and LoanAmount

It is evident that extreme values are present in both ApplicantIncome and LoanAmount, causing both of their distributions to be right-skewed. These extreme values may significantly influence finding patterns and making predications during machine learning. Thus, they need to be treated. To normalise the data, we can perform log transformation on LoanAmount and Income (ApplicantIncome + CoapplicantIncome).

```r
# Data Cleaning
homeloan2$Income <- homeloan2$ApplicantIncome + homeloan2$CoapplicantIncome
homeloan2$LogIncome <- log(homeloan2$Income)
homeloan2$LogLoanAmount <- log(homeloan2$LoanAmount)
```

Figure 8: Log Transformation on Income and LoanAmount

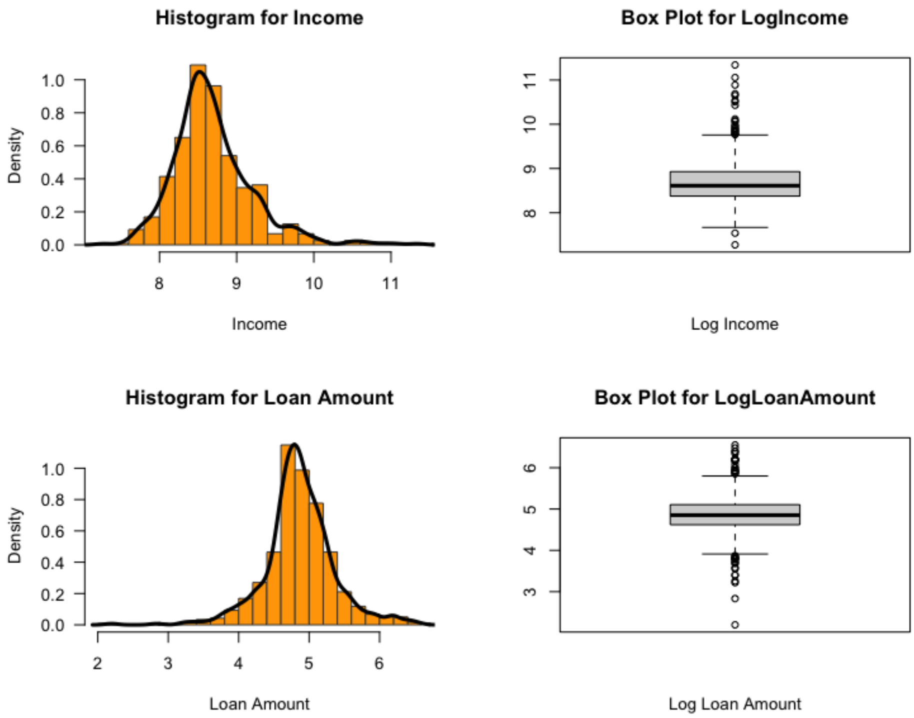

Figure 9: Log Transformation on Income and LoanAmount (Output)

After log transformation, the distribution for Income and LoanAmount is closer to that of a normal distribution. This will improve the accuracy of finding patterns and making predictions when building predictive models as values are no longer skewed.

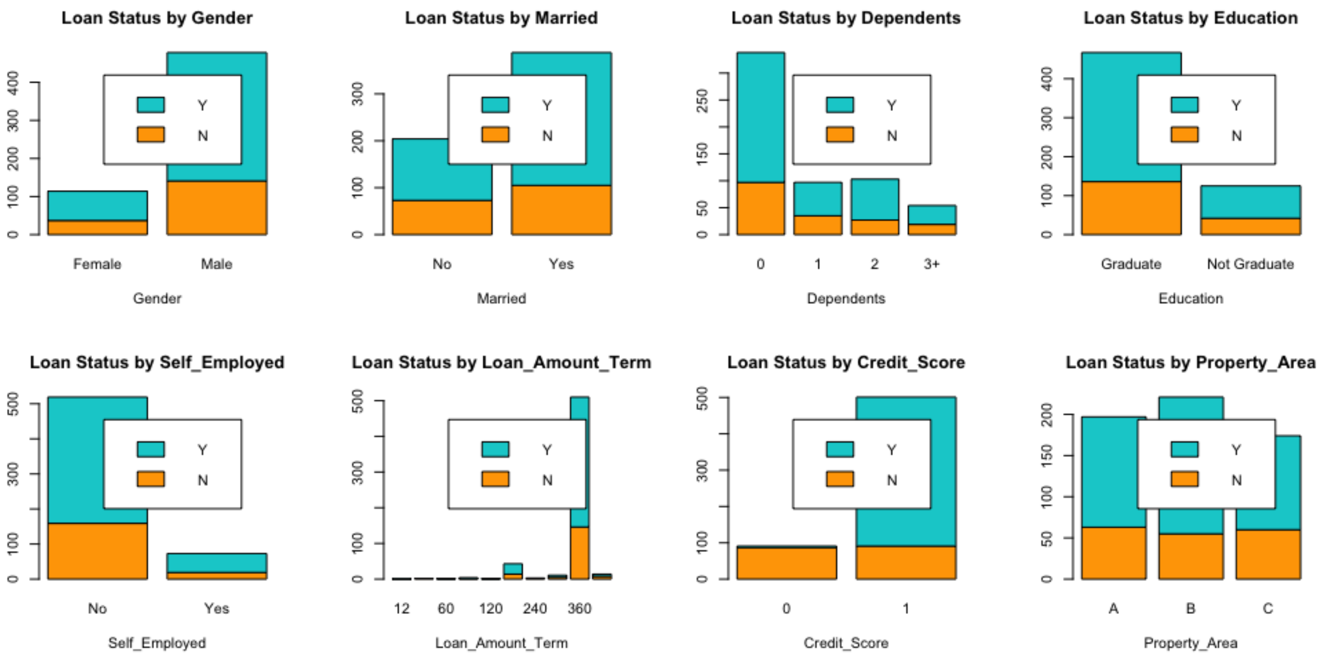

Figure 10: Stacked Barplots for Categorical Variables

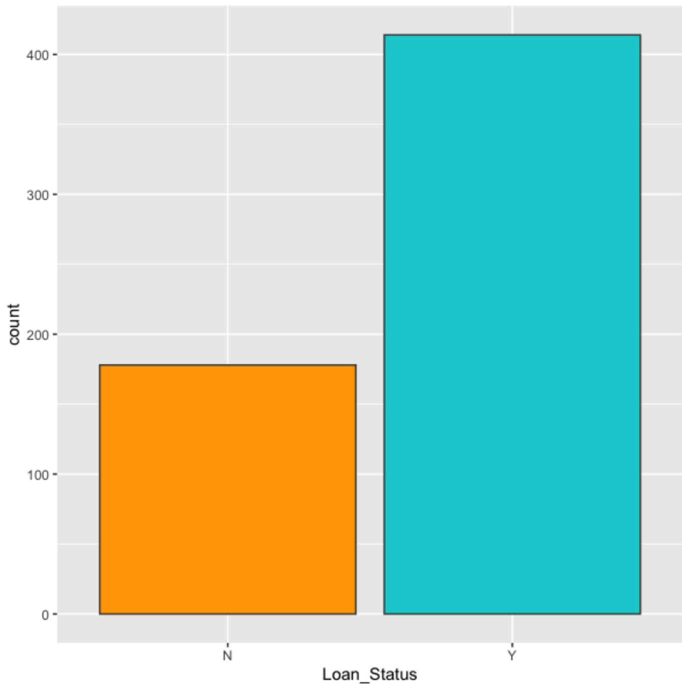

Figure 11: “N” and “Y” count for Loan_Status

1. There are more applications for males and more than half of them have been approved. Meanwhile, despite there being fewer applications for females, more than half of them have been approved as well.
2. Applicants with a bad credit score are significantly less likely to have their loan applications approved.
3. There are more applicants who got their loans approved as compared to those who did not.

```
> nrow(homeloan2)
[1] 592
```

Figure 12: No. of cases in the final cleaned dataset

There are 592 cases in the final cleaned dataset.

Answer to Q3:

(a) No, Loan_ID should not be used as a predictor X variable because it is a unique identifier that will cause overfitting to the model. For example, when run through a classification model such as CART, CART will use Loan_ID to perfectly fit on the trainset, ignoring other variables. This will result in a model that does not provide many insights as it would not be trained to understand general patterns in order to make predictions. Therefore, Loan_ID should not be used as a predictor.

(b)

```
> # Train-Test split -------------------------------------------------------
> train <- sample.split(Y = homeloan2$Loan_Status, SplitRatio = 0.7)
> trainset <- subset(homeloan2, train == T)
> testset <- subset(homeloan2, train == F)
> summary(trainset$Loan_Status)
  N   Y 
125 290 
```

Figure 13: (70:30) split on training data

Firstly, the training data was split with a 70:30 ratio. In the trainset, there is an oversample population of Loan_StatusY, which would affect CART. Hence, there is a need to rebalance the data to make predictions fairer by achieving a ~50% probability of each occurrence.

```
> set.seed(8)
> library(ROSE)
> trainset2 = ovun.sample(Loan_Status~., data=trainset, seed = 8, method ="over", N =
 571)$data
> table(trainset2$Loan_Status)

  Y   N 
290 281 
```

Figure 14: Rebalancing training data

```
Classification tree:
rpart(formula = Loan_Status ~ Gender + Married + Education +
    LogIncome + Credit_Score + LogLoanAmount + Dependents + Self_Employed +
    Property_Area, data = trainset2, method = "class", control = rpart.control(minsplit = 2
0,
    cp = 0))

Variables actually used in tree construction:
[1] Credit_Score  Dependents    Education     Gender        LogIncome
[6] LogLoanAmount Married       Property_Area

Root node error: 281/571 = 0.49212

n= 571

          CP nsplit rel error  xerror     xstd
1  0.4377224      0   1.00000 1.08897 0.042409
2  0.0266904      1   0.56228 0.56228 0.038043
3  0.0195730      6   0.41281 0.51957 0.037098
4  0.0106762     10   0.33452 0.43060 0.034752
5  0.0088968     11   0.32384 0.41281 0.034215
6  0.0083037     13   0.30605 0.39146 0.033537
7  0.0071174     17   0.26335 0.37722 0.033064
8  0.0053381     18   0.25623 0.37722 0.033064
9  0.0035587     20   0.24555 0.38790 0.033420
10 0.0000000     24   0.23132 0.36299 0.032573
```

Figure 15: Output of CART Model

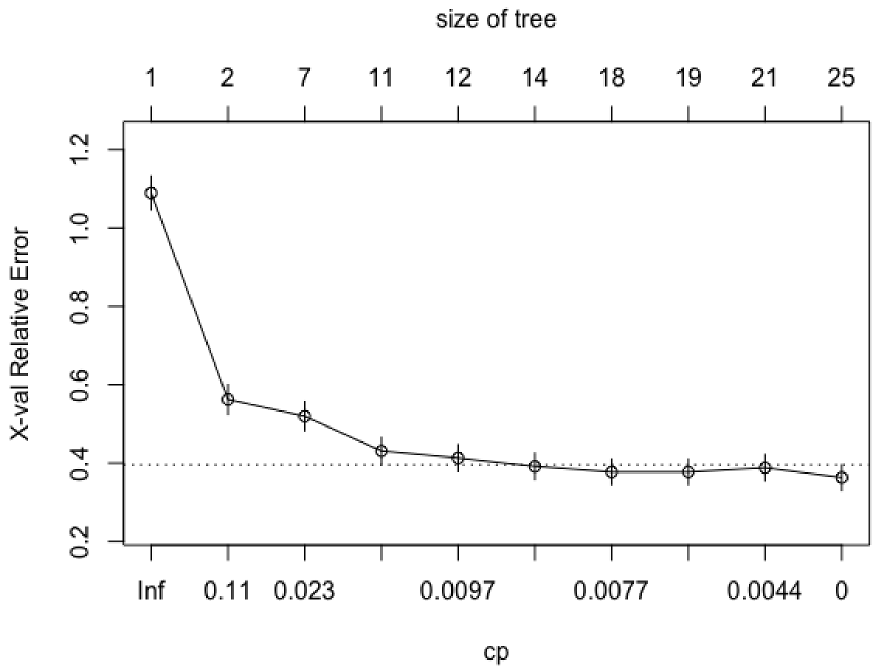

Figure 16: cp plot

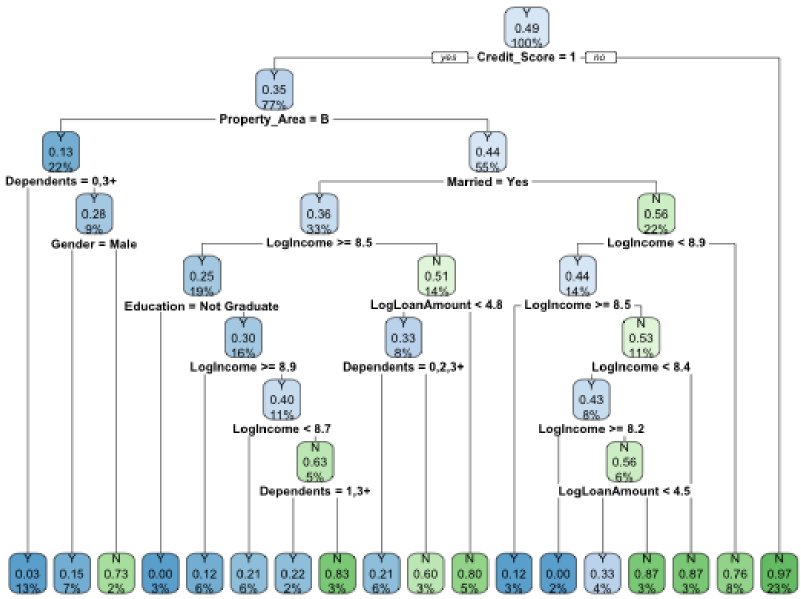

Figure 17: Pruned Tree with cp = 0.0074

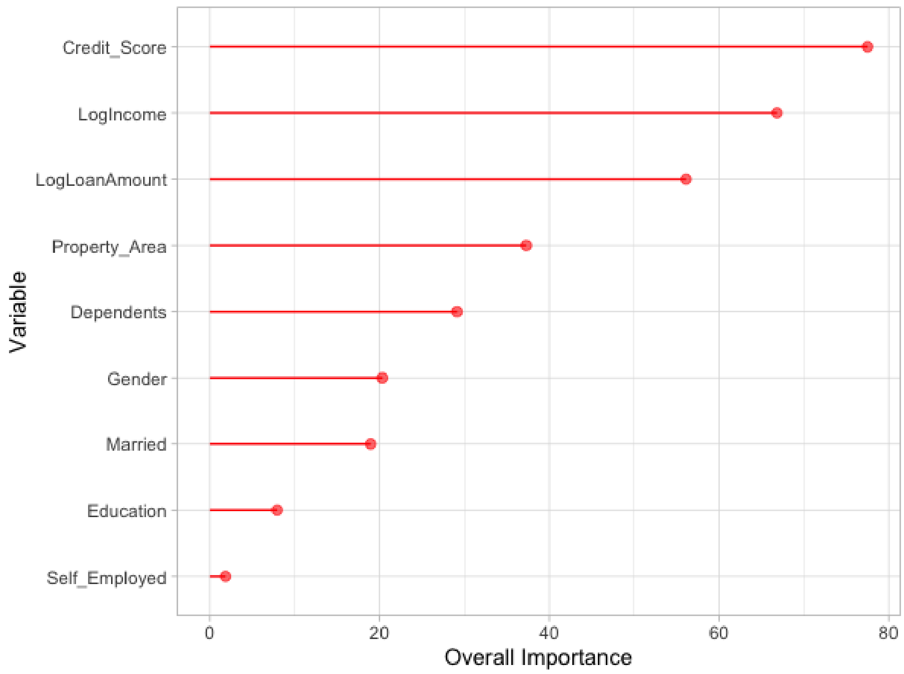

Figure 18: Variable Importance plot for CART

For logistic regression models, unbalanced training data only affects the estimate of the model intercept[2]. Hence, we can use the trainset that was not rebalanced for logistic regression.

[2] https://stats.stackexchange.com/questions/6067/does-an-unbalanced-sample-matter-when-doing-logistic-regression

```
Call:
glm(formula = Loan_Status ~ Gender + Married + Dependents + Education +
    Self_Employed + LogIncome + LogLoanAmount + Credit_Score +
    Property_Area, family = binomial, data = trainset)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.3329  -0.2663   0.4602   0.6573   2.6764

Coefficients:
                      Estimate Std. Error z value           Pr(>|z|)
(Intercept)            -4.4913     2.5612  -1.754            0.07950 .
GenderMale              0.5841     0.3941   1.482            0.13831
MarriedYes              0.4617     0.3569   1.294            0.19578
Dependents1            -0.7169     0.3791  -1.891            0.05864 .
Dependents2            -0.2402     0.4411  -0.545            0.58606
Dependents3+           -0.3140     0.5322  -0.590            0.55514
EducationNot Graduate   0.1938     0.3621   0.535            0.59246
Self_EmployedYes        0.4010     0.4518   0.888            0.37476
LogIncome               0.2836     0.3859   0.735            0.46252
LogLoanAmount          -0.3968     0.3904  -1.017            0.30935
Credit_Score1           4.5923     0.5724   8.023 0.0000000000000103 ***
Property_AreaB          1.1635     0.3806   3.057            0.00224 **
Property_AreaC         -0.1740     0.3235  -0.538            0.59063
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 507.86  on 414  degrees of freedom
Residual deviance: 337.67  on 402  degrees of freedom
AIC: 363.67

Number of Fisher Scoring iterations: 5
```

Figure 18: Logistic Regression Model 1 with AIC = 363.67

```
Call:
glm(formula = Loan_Status ~ Credit_Score + Property_Area, family = binomial,
    data = trainset)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.2071  -0.2760   0.4280   0.7091   2.6141

Coefficients:
               Estimate Std. Error z value            Pr(>|z|)
(Intercept)     -3.2488     0.5810  -5.592 0.000000022492727359 ***
Credit_Score1    4.5013     0.5601   8.037 0.000000000000000923 ***
Property_AreaB   1.0917     0.3690   2.958             0.00309 **
Property_AreaC  -0.1344     0.3081  -0.436             0.66261
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 507.86  on 414  degrees of freedom
Residual deviance: 349.24  on 411  degrees of freedom
AIC: 357.24

Number of Fisher Scoring iterations: 5
```

Figure 19: Logistic Regression Model 2 with AIC = 357.24 after step()

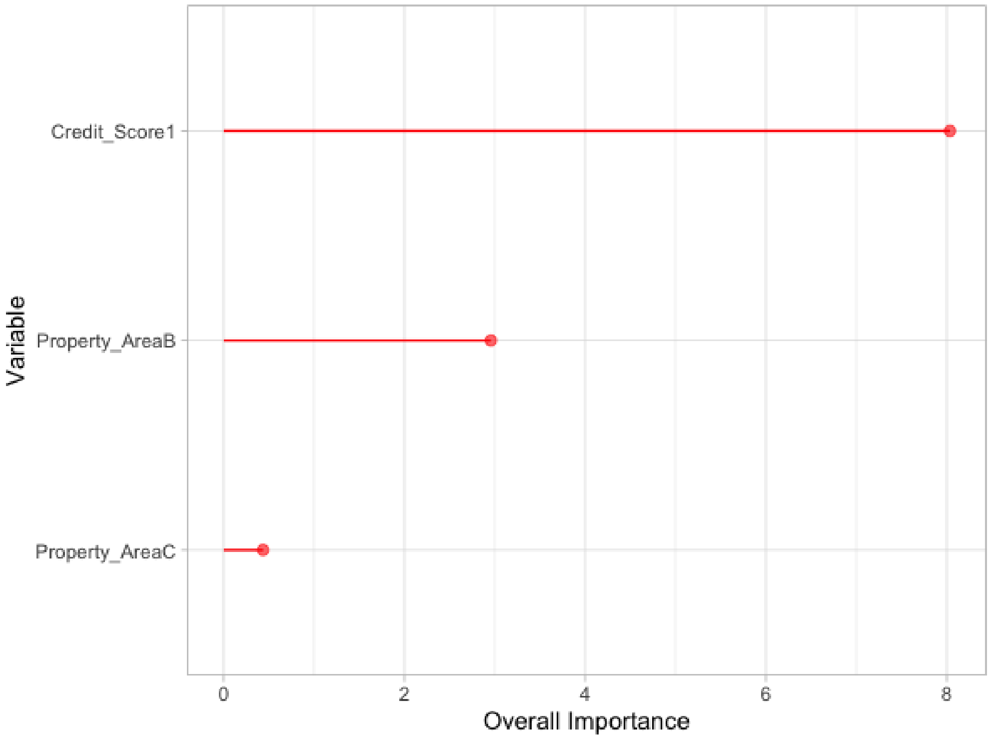

Figure 20: Variable Importance plot for Logistic Regression

	PREDICTED	
ACTUAL	Y	N
N	21	32
Y	108	16

Figure 21: Confusion Matrix for CART

	PREDICTED	
ACTUAL	N	Y
N	26	27
Y	0	124

Figure 22: Confusion Matrix for Logistic Regression

	Accuracy %
CART	79.10
Logistic Regression	84.75

Figure 23: Predictive Accuracy Table

Accuracy for the test sample of CART is 79.10% while accuracy for the test sample of Logistic Regression is 84.75%. Even though the accuracy of Logistic Regression is higher, I found the CART model to be better.

Based on research, key factors that determine a borrower's creditworthiness include capacity. For personal lending, the customer's employment history, current job stability and income amount are all important indicators of the borrower's ability to repay the outstanding debt[3]. Therefore, CART is a better model as it deems other variables such as LogIncome and Self_Employed significant in its prediction while Logistic Regression did not.

```
> # FP for cart
> 21/(21+32) * 100
[1] 39.62264
> # FP for Log Reg
> 27/(26+27) * 100
[1] 50.9434
```

Figure 24: Type 1 Error for CART & Logistic Regression

Moreover, CART has a lower Type 1 error percentage (39.62%) than that of Logistic Regression (50.94%). Therefore, the likelihood of CART wrongly classifying individuals suitable to have their loans approved is lower as compared to Logistic Regression.

(c) The key factor that determines Loan Status is Credit Score, as it has the highest variance importance in both models.

(d) In the case of loan approvals, a Type 1 error is more serious than a Type 2 error because it is a false positive error. This means that an individual is wrongly classified as suitable to take a loan, when in actual fact, he is not suitable to take the loan. This is bad for the bank as they are at a risk of having the outstanding debt to not be repaid.

[3] https://www.forbes.com/advisor/in/personal-loans/top-5-factors-affecting-credit-risk-when-taking-a-personal-loan/

Answer to Q4:

One way to reduce a Type 1 error is to set a lower significance level (α). The probability of a Type 1 error is the same as α, which was set at 0.05 for this case. By changing the α lower than 0.05, it reduces the probability of a Type 1 error and stronger evidence against the null hypothesis is needed before rejecting the null. Hence, if the null hypothesis is true, it is less likely to reject it by chance.

Answer to Q5:

There is no evidence of gender discrimination in loan approved. For this analysis, trainset is used.

```
> summary(trainset$Gender)
Female   Male
    83    332
> gender.sample <- ovun.sample(Gender ~., data= trainset, seed = 8, method ="under")$da
ta
> summary(gender.sample$Gender)
  Male Female
    82     83
```

Figure 25: Rebalancing training data for Gender

To make a fair comparison, the sample for gender was rebalanced to make the gender ratio almost 1:1.

```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)    0.7673     0.2373   3.233  0.00122 **
GenderFemale  -0.1982     0.3294  -0.602  0.54748
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 211.41  on 164  degrees of freedom
Residual deviance: 211.05  on 163  degrees of freedom
AIC: 215.05

Number of Fisher Scoring iterations: 4

> # Return the probability value
> g1$coefficients %>%
+   inv.logit() %>%
+   data.frame()
                      .
(Intercept)  0.6829268
GenderFemale 0.4506213
```

Figure 26: Coefficients from Logistic Regression

Using the rebalanced trainset, I built a Logistic Regression model which includes Gender as the only predictor X variable. In this model, it was found that the p-value of Gender is above the significance level of 0.05 and thus, the null hypothesis should be rejected since it is considered to be statistically insignificant. Moreover, for gender, the model returns a probability a value of 45.1% for a female applicant’s loan to be approved, which is almost 50%. Therefore, trainset tells us that there is no strong evidence of gender discrimination in loan approved.

Answer to Q6:

This analytics problem was limited by the significant percentage (18.92%) of missing values found in the training data, which reduces the accuracy of a model or leads to a biased model. Thus, this will lead to inaccurate predictions. To improve the success of this analytics, the bank can try out **different methods of imputing missing values**, such as KNN and random forest, to see which method will best improve the accuracy of the models.

Banks can also **increase the sample size and number of completed cases** such that when a case is dropped due to outliers or missing values, the effect it has on the training data is negligible since there is a substantial number of cases for the model to use to train in its prediction.

Another way to improve the success of this analytics is to use **multiple algorithms**. Banks should apply all the relevant models, check their performance, and compare them. Just using CART and Logistic Regression may be insufficient as they each have their own limitations.

References

1) Choksi, N. & Joshi, A. (2022, March 19). Top 5 Factors Affecting Credit Risk When Taking A Personal Loan. Forbes Advisor. Retrieved, 23 October 2022, from https://www.forbes.com/advisor/in/personal-loans/top-5-factors-affecting-credit-risk-when-taking-a-personal-loan/
2) conjugateprior (https://stats.stackexchange.com/users/1739/conjugateprior), Does an unbalanced sample matter when doing logistic regression?, URL (version: 2018-11-06): https://stats.stackexchange.com/q/6086
3) Mekala, H. (2018, June 29). Dealing with Missing Data using R. Medium. Retrieved, 23 October 2022, from https://medium.com/coinmonks/dealing-with-missing-data-using-r-3ae428da2d17