Data Wizards (Group 4) Project 2

Di Chen Mai Castellano Tyler Kussee Spencer (Hutchison) Yang

Introduction

In our pursuit of statistical inquiry, we have chosen to explore the Vehicle Loan Default dataset, comprising approximately 41 columns, with one designated as the response variable. Encompassing diverse information, the dataset delves into loan details, including date of birth, employment type, and credit score, alongside loan-related specifics such as disbursal details and loan-to-value ratios. The dataset presents challenges, notably in the form of odd date and time length columns, requiring standardization and transformation into comprehensible formats conducive to model development.

We want to discover the most influential explanatory variables driving loan default, and their impact within the dataset. We also want to find the optimal modeling approach for harnessing the training data, evaluating various methodologies to identify the most effective. Ultimately, our investigation extends to which among them best identifies the underlying dynamics of vehicle loan default prediction.

Obtain/Scrub the data

The data was pulled from the Vehicle Loan Default Prediction datasets available on Kaggle. As mentioned earlier, we have approximately 41 columns, with one of the columns designated as the response variable. First, we'll import the dataset from the training CSV:

We then scrub the unknown values in our length and age fields:

Then we do our calculations and create various other fields for our modeling usage:

Now that we've combed through our dataset, we can now going through our dataset to extract our features and learn the optimal model for classification.

Explore the data

So in the case of this dataset, we need to find the features that have the biggest impact on our "LOAN_DEFAULT" variable. In order to find this, we will be using the XGBoost machine learning algorithm package to train our matrix and label vector from the training data, setting the parameters, and performing cross-validation to find the optimal number of rounds for training the model. Once the parameters and optimal number of rounds were found, we trained the model and then proceeded to calculate the feature importance scores. From there, we select the top 10 features based on their importance to the model, along with plotting the scores to visualize the relative importance of each feature. With these scores and our graph, we found the following features to be the most important:

- LTV
- CURRENT_PINCODE_ID
- PERFORM_CNS_SCORE
- UNIQUEID
- DISBURSED_AMOUNT

- STATE_ID
- SUPPLIER_ID
- PRI_SANCTIONED_AMOUNT
- AGE
- EMPLOYEE_CODE_ID

First, we want to check the correlations between different varibles.

```
colnames(tempTrain)
```

```
##  [1] "UNIQUEID"                              "DISBURSED_AMOUNT"
##  [3] "ASSET_COST"                            "LTV"
##  [5] "BRANCH_ID"                             "SUPPLIER_ID"
##  [7] "MANUFACTURER_ID"                       "CURRENT_PINCODE_ID"
##  [9] "STATE_ID"                              "EMPLOYEE_CODE_ID"
## [11] "MOBILENO_AVL_FLAG"                     "AADHAR_FLAG"
## [13] "PAN_FLAG"                              "VOTERID_FLAG"
## [15] "DRIVING_FLAG"                          "PASSPORT_FLAG"
## [17] "PERFORM_CNS_SCORE"                     "PRI_NO_OF_ACCTS"
## [19] "PRI_ACTIVE_ACCTS"                      "PRI_OVERDUE_ACCTS"
## [21] "PRI_CURRENT_BALANCE"                   "PRI_SANCTIONED_AMOUNT"
## [23] "PRI_DISBURSED_AMOUNT"                  "SEC_NO_OF_ACCTS"
## [25] "SEC_ACTIVE_ACCTS"                      "SEC_OVERDUE_ACCTS"
## [27] "SEC_CURRENT_BALANCE"                   "SEC_SANCTIONED_AMOUNT"
## [29] "SEC_DISBURSED_AMOUNT"                  "PRIMARY_INSTAL_AMT"
## [31] "SEC_INSTAL_AMT"                        "NEW_ACCTS_IN_LAST_SIX_MONTHS"
## [33] "DELINQUENT_ACCTS_IN_LAST_SIX_MONTHS" "AVERAGE_ACCT_AGE"
## [35] "CREDIT_HISTORY_LENGTH"                 "NO_OF_INQUIRIES"
## [37] "LOAN_DEFAULT"                          "AGE"
## [39] "SELF_EMPLOYED"                         "SALARIED"
## [41] "NULL_EMPLOYMENT"
```

```
# Convert all columns to numeric
tempTrain[] <- lapply(tempTrain, as.numeric)

# Check the structure of the 'train' dataset to verify numeric conversion
str(tempTrain)
```

```
## 'data.frame':    233154 obs. of  41 variables:
##  $ UNIQUEID                          : num  420825 537409 417566 624493 539055 ...
##  $ DISBURSED_AMOUNT                  : num  50578 47145 53278 57513 52378 ...
##  $ ASSET_COST                        : num  58400 65550 61360 66113 60300 ...
##  $ LTV                               : num  89.5 73.2 89.6 88.5 88.4 ...
##  $ BRANCH_ID                         : num  67 67 67 67 67 67 67 67 67 67 ...
##  $ SUPPLIER_ID                       : num  22807 22807 22807 22807 22807 ...
##  $ MANUFACTURER_ID                   : num  45 45 45 45 45 45 45 45 45 45 ...
##  $ CURRENT_PINCODE_ID                : num  1441 1502 1497 1501 1495 ...
##  $ STATE_ID                          : num  6 6 6 6 6 6 6 6 6 6 ...
##  $ EMPLOYEE_CODE_ID                  : num  1998 1998 1998 1998 1998 ...
##  $ MOBILENO_AVL_FLAG                 : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ AADHAR_FLAG                       : num  1 1 1 1 1 1 1 1 1 0 ...
##  $ PAN_FLAG                          : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ VOTERID_FLAG                      : num  0 0 0 0 0 0 0 0 0 1 ...
```

```
##  $ DRIVING_FLAG                        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ PASSPORT_FLAG                       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ PERFORM_CNS_SCORE                   : num  0 598 0 305 0 825 0 17 718 818 ...
##  $ PRI_NO_OF_ACCTS                     : num  0 1 0 3 0 2 0 1 1 1 ...
##  $ PRI_ACTIVE_ACCTS                    : num  0 1 0 0 0 0 0 1 1 0 ...
##  $ PRI_OVERDUE_ACCTS                   : num  0 1 0 0 0 0 0 0 0 0 ...
##  $ PRI_CURRENT_BALANCE                 : num  0 27600 0 0 0 ...
##  $ PRI_SANCTIONED_AMOUNT               : num  0 50200 0 0 0 ...
##  $ PRI_DISBURSED_AMOUNT                : num  0 50200 0 0 0 ...
##  $ SEC_NO_OF_ACCTS                     : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ SEC_ACTIVE_ACCTS                    : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ SEC_OVERDUE_ACCTS                   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ SEC_CURRENT_BALANCE                 : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ SEC_SANCTIONED_AMOUNT               : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ SEC_DISBURSED_AMOUNT                : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ PRIMARY_INSTAL_AMT                  : num  0 1991 0 31 0 ...
##  $ SEC_INSTAL_AMT                      : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ NEW_ACCTS_IN_LAST_SIX_MONTHS        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ DELINQUENT_ACCTS_IN_LAST_SIX_MONTHS: num  0 1 0 0 0 0 0 0 0 0 ...
##  $ AVERAGE_ACCT_AGE                    : num  0 1.92 0 0.67 0 1.75 0 0.17 4.67 1.58 ...
##  $ CREDIT_HISTORY_LENGTH               : num  0 1.92 0 1.25 0 2 0 0.17 4.67 1.58 ...
##  $ NO_OF_INQUIRIES                     : num  0 0 0 1 1 0 0 0 1 0 ...
##  $ LOAN_DEFAULT                        : num  1 2 1 2 2 1 1 1 1 1 ...
##  $ AGE                                 : num  35 33 33 25 41 28 30 29 27 50 ...
##  $ SELF_EMPLOYED                       : num  0 1 1 1 1 1 0 0 1 0 ...
##  $ SALARIED                            : num  1 0 0 0 0 0 1 1 0 1 ...
##  $ NULL_EMPLOYMENT                     : num  0 0 0 0 0 0 0 0 0 0 ...
```

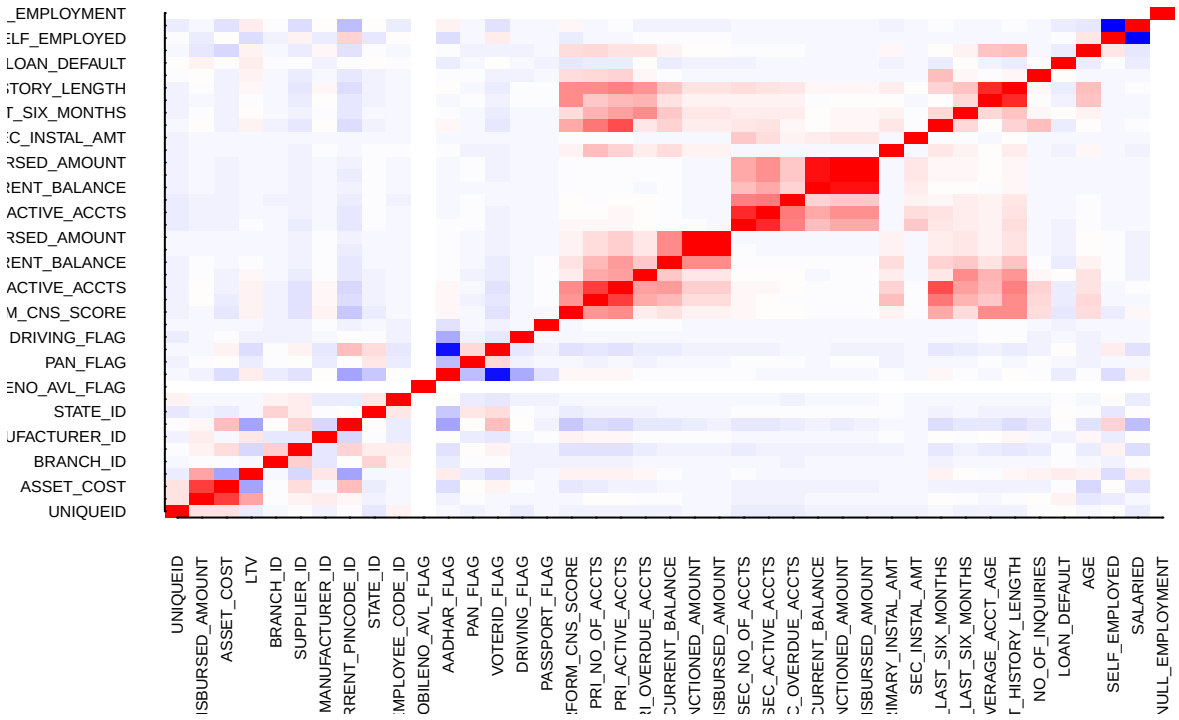
```
# Check for NaN or infinite values in the correlation matrix and replace with 0
correlation_matrix <- cor(tempTrain)
```

```
## Warning in cor(tempTrain): the standard deviation is zero
```

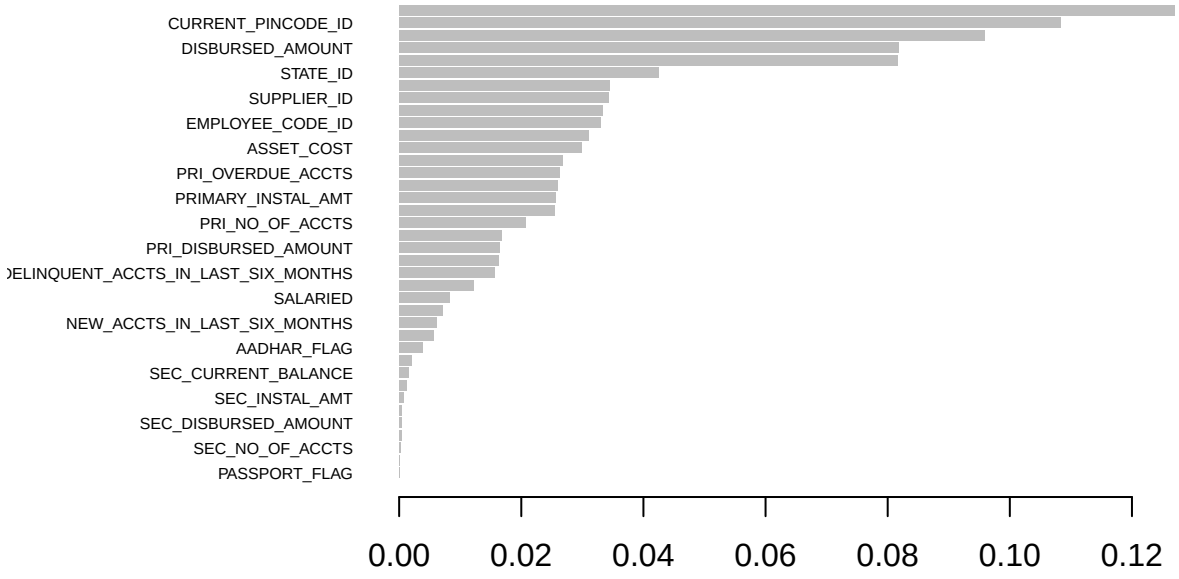
```
correlation_matrix <- as.matrix(correlation_matrix)
correlation_matrix[is.nan(correlation_matrix) | is.infinite(correlation_matrix)] <- 0

# Plot correlation matrix directly without clustering with variable labels
image(1:nrow(correlation_matrix), 1:ncol(correlation_matrix), correlation_matrix,
      main = "Correlation Matrix Heatmap",
      xlab = "",
      ylab = "",
      col = colorRampPalette(c("blue", "white", "red"))(100),
      axes = FALSE)

# Add labels to the axes with smaller font size and remove numbers
axis(1, at = 1:ncol(correlation_matrix), labels = colnames(correlation_matrix), las = 2, cex.axis = 0.5, tcl
axis(2, at = 1:nrow(correlation_matrix), labels = rownames(correlation_matrix), las = 2, cex.axis = 0.5, tcl
```



With all of this in mind, we're able to now subset the training dataset with these important features, and split it into new training and testing datasets. we can start building our classifiers and getting results.



Model the data

Interpret the data

Obstacles

Initially, we allocated individual tasks to each team member and emphasized focusing on their assigned responsibilities. However, we noticed variations in the approach to problem-solving among team members. During collaboration on this project using Git, conflicts arose when pushing changes to the main branch. Subsequently, we reached an agreement stipulating that each team member must have their changes reviewed and approved by the next person before merging them into the main branch.

Conclusion

Appendix