Helping Local Entrepreneurs Secure Funding

Team: Data Geek
Arjun Talwar, Harish G, Surabhi Agarwal,
Takashi Nishizu, Yuto Saiki

Summary

Kiva is a micro-financing firm that aims to source funding for local entrepreneurs at zero interest. The biggest challenge with crowdsourcing is that it has always been a black box when it comes to determining the success of raising funds. Unlike traditional banks, where there are clear criteria that go into loan approval, there could be a plethora of factors that determine the success of crowdsourcing. 90% of the local entrepreneurs seek micro-financing funds between $125 to $700. Banks are not going to entertain these fund seekers. Their only cheap option tends to be microfinancing via crowdsourcing through donations. We would like to analyze the data of Kiva crowdsourcing loan data to mine for critical factors that influence fundraising. We aim to provide a platform that would enable local entrepreneurs to not only check if they are likely to raise funds but also highlight the important features that would increase their chance of fundraising through donations.

The data from Kiva contained information from over 80 different countries. Our group decided to just focus on the Philippines for this project as we had the most data points from that country. From a business point of view, the Philippines has one of the highest demand for micro-financing in a diverse range of sectors. We included various metrics such as gender, date requested, activity, sector, and requested loan amount as potential factors that would affect fundraising. As we wanted to explore many different algorithms, all input fields were quantized. The dates were converted into quarters and the loan amount was divided into 4 logical buckets (Low, Medium Low, Medium High, and High). An issue with the data we faced was that it was skewed toward successful funding. Over 95% of the data points consisted of successes. Hence, we applied the Synthetic Minority Oversampling TEchnique (SMOTE) to get a 50% baseline rate for successful data points vs unsuccessful data points. With the data prepared, our group trained 4 different algorithms and felt that the **Decision Tree algorithm** was ideal as we placed emphasis on **specificity**. We wanted to correctly identify folks who would **not get a loan** and potentially give them feedback on how they can improve their chances.

The **Decision Tree algorithm** provides an accuracy of 90.1% for a baseline rate of 50% after SMOTE. The critical factors that determine the success of funding are **Applying for the loan in Q1, Keeping the loan amount more than 500 php, Having a monthly repayment scheme, and applying as a single person.** Funds requested in the **retail sector** showed to have a higher chance of securing funding as well.

While these results provide a good understanding of the critical factors that determine the success of microfinancing, there is a lot of scope for future work. Incorporating data from other micro-financing firms such as 51Give and Grameen Bank would allow us to direct borrowers to the right firm based on their profile. A lot of creative ideas stem from local entrepreneurs that often fail to be implemented due to lack of funding. We strongly believe that our platform would not only enable many entrepreneurs to achieve their dreams but also significantly impact numerous communities.

Detailed Report

Problem description

Here, our business goal is to provide a platform for local entrepreneurs to verify their funding ability when borrowing. Therefore, stakeholders for our goal are mainly local entrepreneurs who hope to improve their business with funding. However, there is a challenge to be overcome. Firstly, many entrepreneurs are unaware of their ability to raise funds through crowdsourcing platforms because they do not have the information to identify critical factors that affect crowdfunding decisions. At this point, our analysis will be quite helpful since we will make those factors clear through a dataset by Kiva.

Data description

The dataset we used is kiva funding data from 2014 to 2017. We can get it for free. The model we implemented is predictive and supervised. Output variables are binary (fully funded or not) and there are 33 input variables (you can see all of the variables below). In addition, a sample of five rows is as follows.

[Output variables]

Funded_bracket (fully funded or not)

[Input variables]

L_req, ML_req, MH_req, H_req, sector_retail, sector_food, sector_clothing, sector_entertainment, sector_services, sector_transportation, sector_education, sector_arts, sector_health, sector_manufacturing, sector_housing, sector_wholesale, sector_personaluse, sector_agriculture, sector_construction, material, buy, purchase, requested_month, requested_in_q1, requested_in_q2, requested_in_q3, requested_in_q4, term_in_months, male_borrower, female_borrower, combo_borrowers, Irregular_repayment, monthly_repayment, bullet_repayment

[A sample of five rows]

funded_bracket	L_req	ML_req	MH_req	H_req	sector_retail	sector_food	sector_clothing	sector_entertainment	sector_services
1	1	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0

sector_transportation	sector_education	sector_arts	sector_health	sector_manufacturing	sector_housing	sector_wholesale	sector_personaluse	sector_agriculture	sector_construction	material	buy	purchase
0	0	0	0	0	1	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	0	0	0	1	0	0	1	0
0	0	0	0	0	0	0	0	1	0	0	0	1

requested_in_q1	requested_in_q2	requested_in_q3	requested_in_q4	term_in_months	male_borrower	female_borrower	combo_borrowers	Irregular_repayment	monthly_repayment	bullet_repayment
0	1	0	0	3	0	1	0	1	0	0
1	0	0	0	8	0	1	0	1	0	0
1	0	0	0	8	0	1	0	1	0	0
1	0	0	0	8	0	1	0	1	0	0
0	0	1	0	8	0	1	0	1	0	0

To find where we should focus on the dataset, we overviewed it by subtracting some features, such as region and disbursed time. Finally, we decided to choose a specific country, Philippines, as a practical dataset because it includes about 24.5% of all the dataset.

Data Preparation

Regarding data preparation, we conducted the following processes: (1) text mining, (2) one-hot coding, (3) merging with the complement data, (4) aggregating columns, (5) categorizing the loan amount requested, (6) handling of N/A, (7) downsizing, and (8) preprocess for SMOTE (Synthetic Minority Oversampling Technique), by mainly Excel and R.

1. This dataset included text data in uses and tags. While we just eliminated tags columns since words are too diversified there, we split all sentences into words on the ‘uses’ columns. After splitting into each word, we executed word clouds and choose the three most frequent words per sector on the assumption that different sectors might have different keywords.

2. We have many categorical data in the extracted word, sector, gender, and so on. We implemented one-hot coding for them by converting one column including multiple variables into multiple binary columns which can be dealt with machine learning. For example, we converted columns with five countries, into five binominal columns: if the countries is Philippine, only Philippine's column returns 1 whilst the other four columns return 0.
3. Even though the dataset held MPI (Multidimensional Poverty Index) data, many cells were empty. We complimented that dataset by combining with MPI data originally provided by the United Nations.
4. Some features were aggregated to avoid the excessively minute and noisy results. For instance, the original data has the time stamp on the YYYY-MM-DD HH:MI:Sec format. We transformed that time data to 4 categories based on the quarter: Q1 which is Jan-Mar. Most importantly, we converged the continuous y-variable (the loan amount) into binary data, fully-funded or not, owing to the skewness of this data, which we will mention in (7).
5. The requested loan amount was divided into 4 categories. Low (<200 php), Medium-Low (200php - 350 php), Medium-High (350 php - 500 php), and High (>500 php)
6. We simply deleted rows which have N/A or obviously mistakenly-inputted cells. We are convinced that it doesn't matter because we still have a large amount of data.
7. After the preparation mentioned above, we deleted some redundant columns indicating the same features, such as country name and country code, and pruned uncontrolled variables, such as ID number and the month when the borrowers actually gained money. Nevertheless, that dataset was still too large, 671k rows times 90 columns. It was infeasible to deal for some machine learning methods, especially decision tree, in R. Hence, we decided to downsize the number of rows into 160k by focusing Philippines, which consists 25 % of all funding, as our focused market, considering that different countries should have different results due to various culture and different currencies. We decreased the number of columns to 30 as well, by eliminating some features, such as "secter_keyword" columns as they were very data-consuming.
8. Finally, we tried the preprocess for SMOTE on this dataset to deal with the imbalance of y-variables: 95% sample is fully funded. We maintained all not-fully-funded rows, and randomly picked out the same number of rows from fully-funded rows as that of not-fully-funded rows in order to make the training more meaningful.

Solution: Methods Applied and Evaluation Results

We decided to apply four different methods to predict whether our local entrepreneur will get the loan or not. As part of the different methods, we also controlled for the amount of loan (in PHP) that he/she/they requested. We chose the following 4 methods:

1. Decision tree
2. Logistic Regression
3. Naive Bayes
4. Support Vector Machine (SVM)

We ran them on the Dataset which we had prepared and were surprised to see very high values for all the algorithms. A deeper analysis revealed that our Base rate (i.e. the amount of bias in the dataset) was very huge i.e around 96% of the data comprised of loan requests that were approved. This prompted us to re-evaluate our dataset and reconfigure it to perform a SMOTE (Synthetic Minority OverSampling technique: an approach to the construction of classifiers from imbalanced dataset wherein we took randomly sampled and replicated our minority test cases (rows) to remove the inherent bias with the dataset). For most of the

algorithms described below: We decided to use 5 fold cross-validation to train our model well and eliminating possible over-fitting which is a common problem with decision trees. We used a 95% significance level to determine our confidence interval. Once the algorithm was run, we predicted the variable importance as well which showed us what matters in the application the local entrepreneur fills. [Please see appendix for output plots, ROC curves, confusion matrices, et. al for all of these algorithms] Detailed results, before SMOTE and after SMOTE are provided below:

[Before SMOTE]

	Decision Tree	Logistic Regression	Naive Bayes Model	Support Vector Machine
Accuracy	98.18%	98.17%	80.5%	92.6%
Precision	40.7%	43.18%	95.44%	1.8%
Recall	1.9%	3.3%	65.11%	98.725%
FP	567	559	202	7

[After SMOTE]

	Decision Tree	Logistic Regression	Naive Bayes Model	Support Vector Machine
Accuracy	89.65%	89.5%	47.75%	70.3%
Precision	86.18%	88.7%	0%	63.8%
Recall	94.07%	91.19%	100%	100%
FP	33	51	591	0

Conclusions/Recommendations

Since we value local entrepreneurs and want as many of them to secure a loan, we want to reduce as many false positives as possible, giving everyone a loan who deserves. Also, we want to make a balance between Accuracy, Precision, and Recall. Hence, we are going to select the Decision Tree as our model. The important features that we identified from the decision tree are shown in the picture on the right. Additionally, the area under ROC curve is 0.899 which is one of the highest accuracy among all the algorithms we ran after SMOTE. Some of the business that we can gather from this exercise are stated below:

1. It is beneficial to request for a loan in quarter 1 (i.e. the first 3 months of the year) rather than requesting it in other 3 quarters.
2. It is advisable to apply for loans with an amount greater than 500 PHP; although this requires further causal analysis, one potential reason why such loans are accepted could be because loans having higher amounts have better business ideas.
3. The terms in months (Number of months the loan is taken for) is an important feature that needs to be considered when applying for the fund.
4. It is better to apply as a single person or in a group with a homogeneous composition (either all males or all females) rather than in a heterogeneous group (having a mixture of males and females)
5. Clearly, three sectors were the most favorable for securing a loan. These were - Agriculture, Retail and Service. There is a lesser probability to secure a loan if it involves other sectors.
6. The monthly repayment is preferred over irregular payments and 1 shot payment, which is intuitive as it reduces the risks but is a valuable insight nevertheless since this is an important topic of consideration for several local entrepreneurs while applying for the loans.

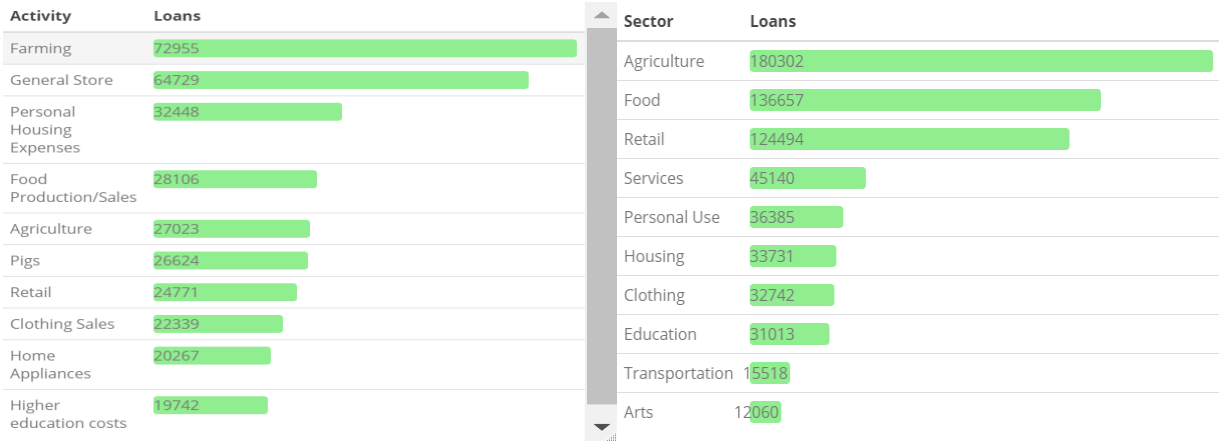
Our future plan is to expand on this dataset and create an app, which will tell the loan borrowers what they can improve in their loan application as well as with their current application how much they can get. We also want to expand this idea for other financial institutions. We can add more data collected from different institutions to help loan borrowers to understand from where they can get a loan with their current state.

Appendix:

Our Choice of Algorithms

1. Classification regression trees or Decision trees have several advantages which is why we decided to use them. Firstly, they implicitly perform variable screening or feature selection. Non-linear relationships between parameters do not affect the tree performance. Another main reason was that decision trees are very easy to interpret and explain and aren't like the typical black box algorithms like Neural networks which are hard to explain.
2. Logistic regression is used to predict an outcome variable that is in a categorical form, predictor variables that are either continuous or in categorical form. This was exactly our case as many of our predictor variables were categorical. Even our output variable was a categorical variables. These, like decision trees, are easy to implement and interpret.
3. Naive Bayes is also used to predict the probability of different classes based on various attributes. This algorithm is mostly used in text classification and with problems having multiple classes. Therefore, we picked this as well.
4. Support Vector Machine is a supervised machine learning algorithm which can be used for classification/regression problems. It can efficiently perform even a nonlinear classification using a kernel trick, implicitly mapping inputs into a high dimensional feature space. Therefore, we decided to use this as well.

Exploratory Data Analysis



Activity	Loans
Farming	72955
General Store	64729
Personal Housing Expenses	32448
Food Production/Sales	28106
Agriculture	27023
Pigs	26624
Retail	24771
Clothing Sales	22339
Home Appliances	20267
Higher education costs	19742

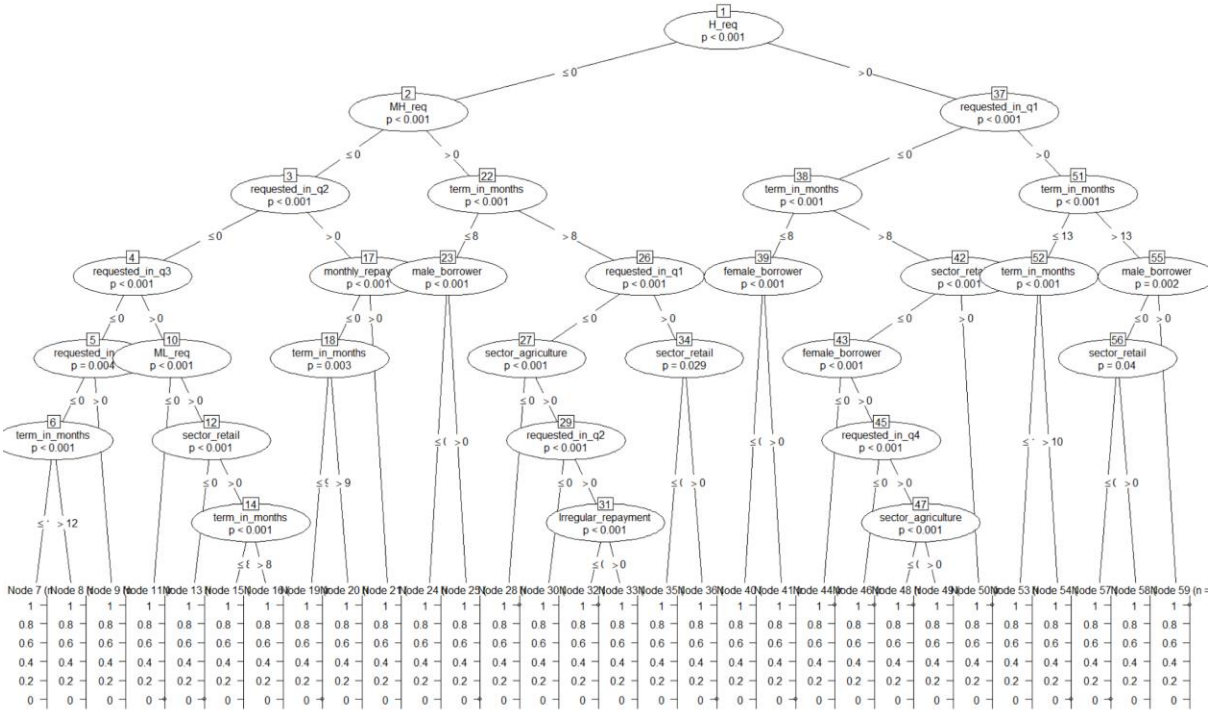
Sector	Loans
Agriculture	180302
Food	136657
Retail	124494
Services	45140
Personal Use	36385
Housing	33731
Clothing	32742
Education	31013
Transportation	15518
Arts	12060

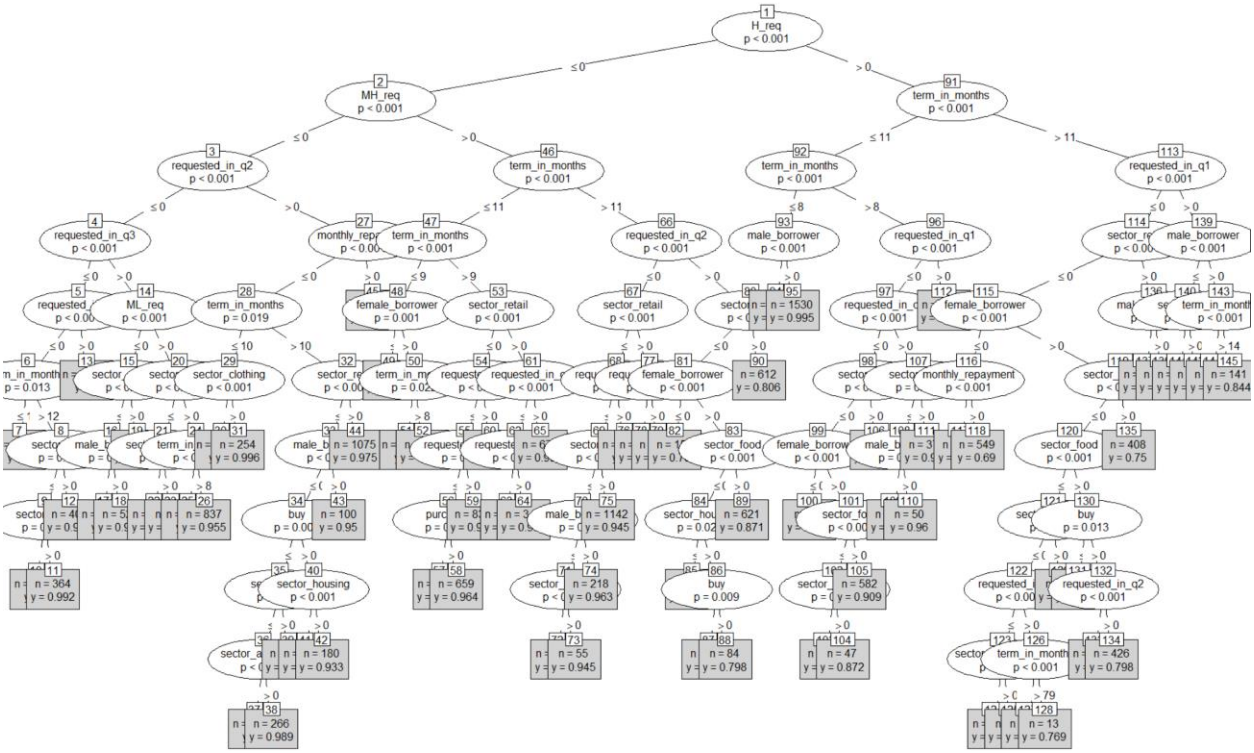
Countries	Loans
Philippines	160441
Kenya	75825
El Salvador	39875
Cambodia	34836
Pakistan	26857
Peru	22233
Colombia	21995
Uganda	20601
Tajikistan	19580
Ecuador	13521

Before SMOTE Analysis

Decision Trees

```
>   cvtree
Conditional Inference Tree

160360 samples
    33 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 128288, 128288, 128288, 128288, 128288
Resampling results:

  RMSE       Rsquared   MAE
  0.1205897  0.1603439  0.0289352

Tuning parameter 'mincriterion' was held constant at a value of 0.95
```





only 20 most important variables shown (out of 33)

	Overall
H_req	100.0000
term_in_months	74.0529
monthly_repayment	35.3038
requested_in_q1	34.9091
Irregular_repayment	34.8552
L_req	26.4799
female_borrower	25.2665
male_borrower	25.2665
ML_req	21.7415
sector_retail	9.8744
requested_in_q3	9.2088
requested_in_q2	9.1150
sector_agriculture	6.5639
sector_housing	2.1543
buy	2.0411
purchase	1.8405
sector_arts	1.1728
sector_manufacturing	0.7787
sector_services	0.7156
sector_education	0.6692

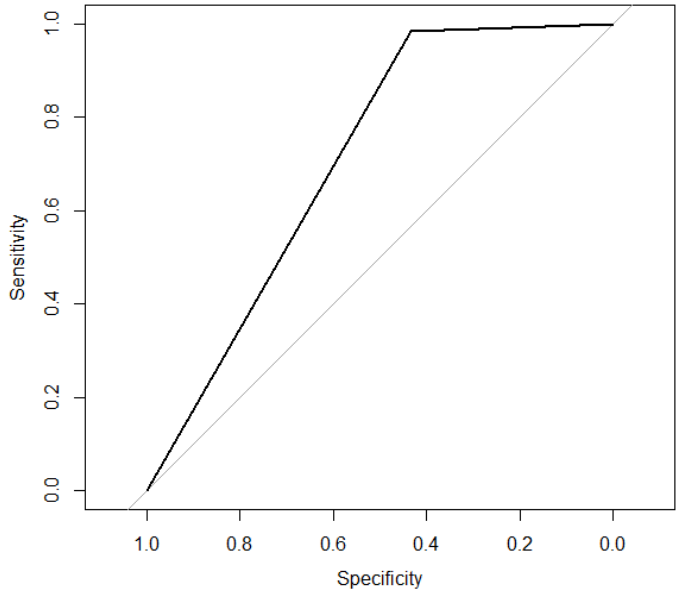
Warning message:

```
>   print(metrics)
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0     0     0
         1   558 31514

               Accuracy : 0.9826
                 95% CI : (0.9811, 0.984)
    No Information Rate : 0.9826
    P-Value [Acc > NIR] : 0.5113

                  Kappa : 0
 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.0000
            Specificity : 1.0000
         Pos Pred Value :    NaN
         Neg Pred Value : 0.9826
             Prevalence : 0.0174
         Detection Rate : 0.0000
   Detection Prevalence : 0.0000
      Balanced Accuracy : 0.5000

       'Positive' Class : 0
```

Naive Bayes

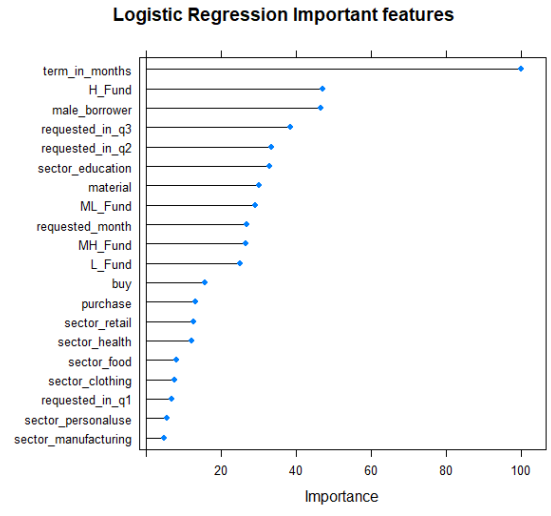
```
>   print(naive.bayes.imp)
loess r-squared variable importance

  only 20 most important variables shown (out of 34)

                    Overall
H_Fund             100.0000
term_in_months      83.5704
monthly_repayment   39.3888
Irregular_repayment 38.8311
requested_in_q1     37.4422
female_borrower     29.3909
male_borrower       29.3909
ML_Fund             28.7381
L_Fund              22.6329
requested_month     19.3230
requested_in_q3     11.0175
sector_retail       10.8643
requested_in_q2      8.6943
MH_Fund              7.8475
sector_agriculture   6.7681
buy                  2.5031
sector_housing       2.0135
purchase             1.9085
sector_arts          1.2339
sector_services      0.8105
```

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 377  18
         1 202 534

               Accuracy : 0.8055
                 95% CI : (0.7812, 0.8282)
    No Information Rate : 0.5119
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6137

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.6511
            Specificity : 0.9674
         Pos Pred Value : 0.9544
         Neg Pred Value : 0.7255
             Prevalence : 0.5119
         Detection Rate : 0.3333
   Detection Prevalence : 0.3492
      Balanced Accuracy : 0.8093

       'Positive' Class : 0
```




```
>   roc(pred.int, as.integer(test_dataset$funded_bracket) , plot=TRUE,levels=base::levels(as.factor(pred)))

Call:
roc.default(response = pred.int, predictor = as.integer(test_dataset$funded_bracket),     levels = base::levels(as.factor(pred)), plot = TRUE)

Data: as.integer(test_dataset$funded_bracket) in 23 controls (pred.int 0) < 32049 cases (pred.int 1).
Area under the curve: 0.7086
```

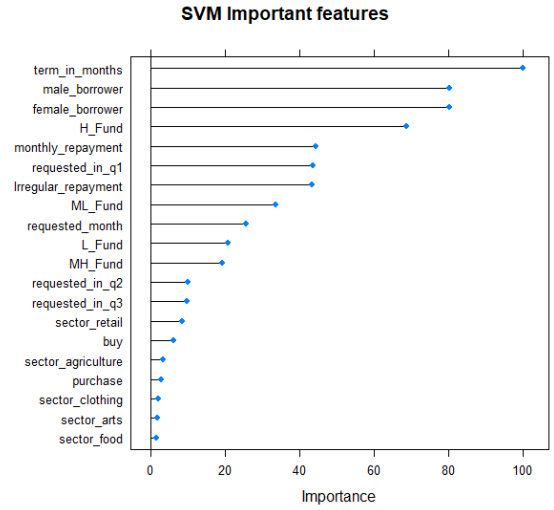
Logistic Regression



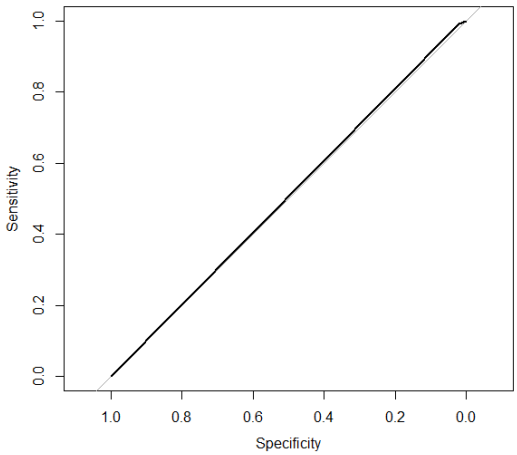

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0    19    25
         1   559 31469

               Accuracy : 0.9818
                 95% CI : (0.9803, 0.9832)
    No Information Rate : 0.982
    P-Value [Acc > NIR] : 0.6099

                  Kappa : 0.0587

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.0328720
            Specificity : 0.9992062
         Pos Pred Value : 0.4318182
         Neg Pred Value : 0.9825465
             Prevalence : 0.0180220
         Detection Rate : 0.0005924
   Detection Prevalence : 0.0013719
      Balanced Accuracy : 0.5160391

       'Positive' Class : 0
```

```
Call:
roc.default(response = as.numeric(glm.pred), predictor = as.numeric(test_dataset$funded_bracket,     plot = TRUE))

Data: as.numeric(test_dataset$funded_bracket, plot = TRUE) in 57 controls (as.numeric(glm.pred) 1) < 32015 cases (as.numeric(glm.pred) 2).
Area under the curve: 0.6496
```

SVM



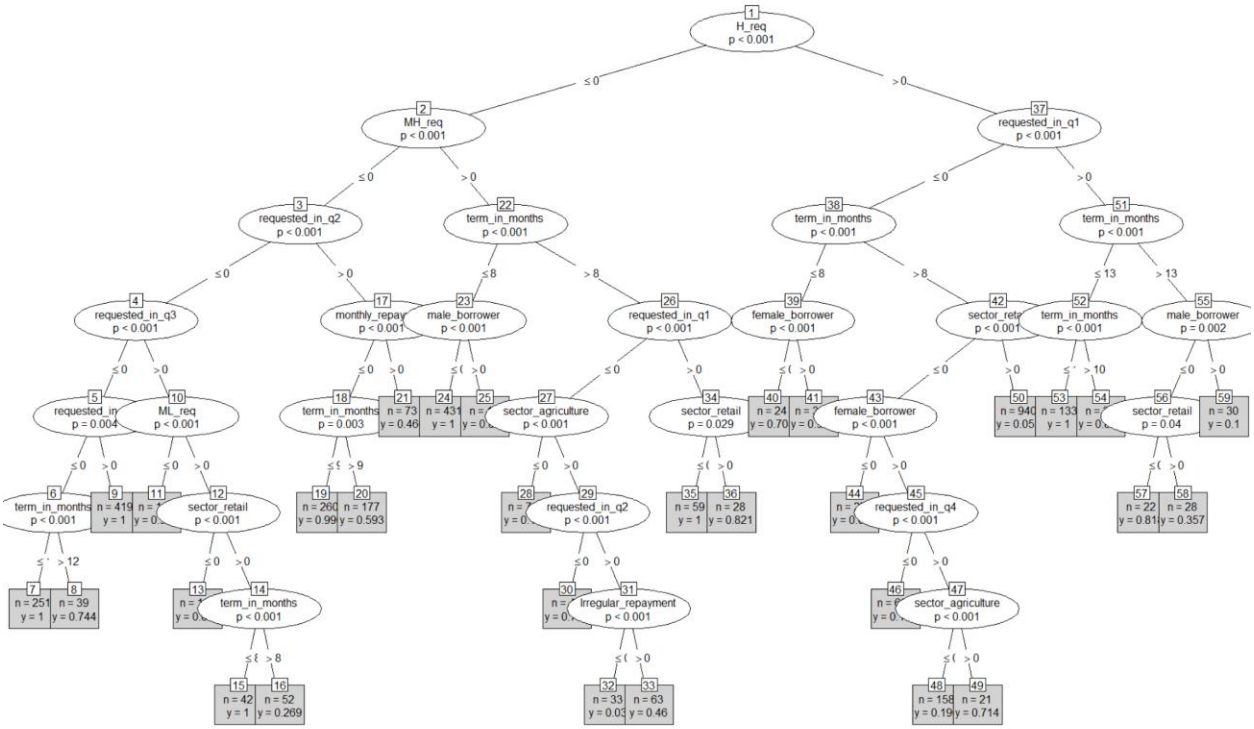

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0  139 6553
         1   11 1297

               Accuracy : 0.1795
                 95% CI : (0.1711, 0.1881)
    No Information Rate : 0.9812
    P-Value [Acc > NIR] : 1

                  Kappa : 0.0041

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.92667
            Specificity : 0.16522
         Pos Pred Value : 0.02077
         Neg Pred Value : 0.99159
             Prevalence : 0.01875
         Detection Rate : 0.01738
   Detection Prevalence : 0.83650
      Balanced Accuracy : 0.54594

       'Positive' Class : 0
```

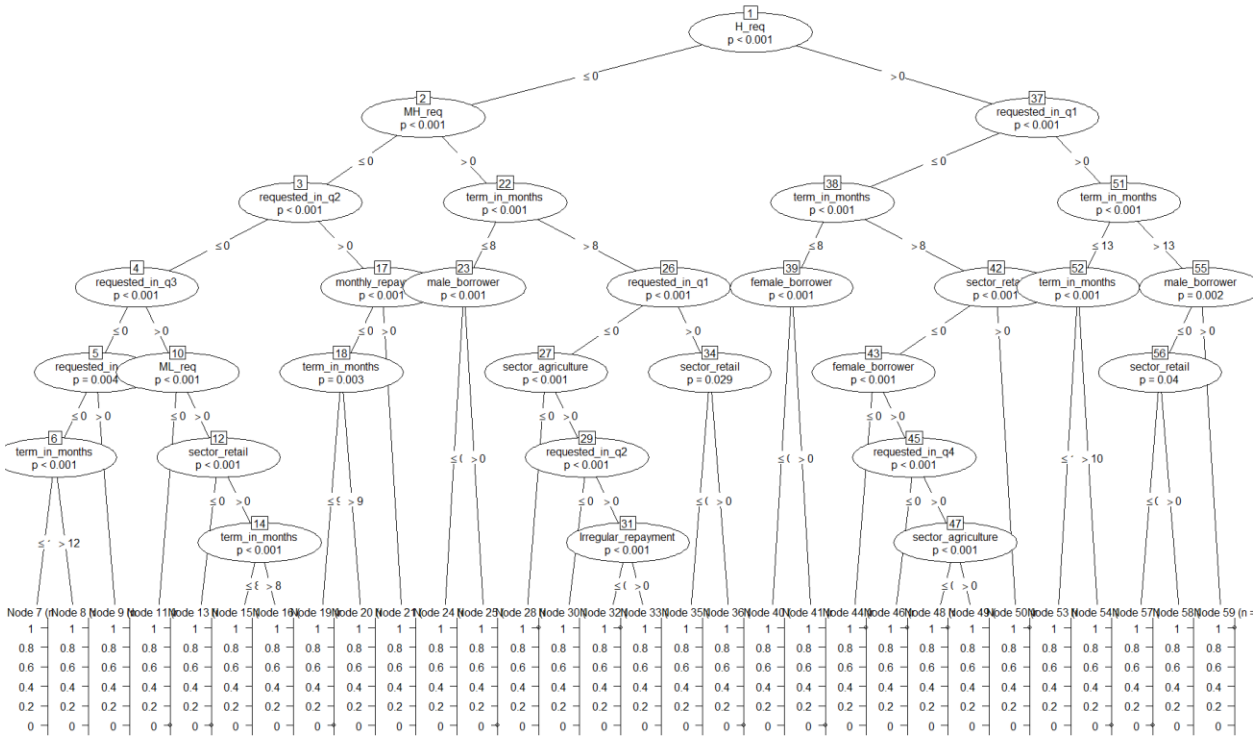



```
Call:
roc.default(response = as.integer(svm.pred), predictor = as.integer(test_dataset1$funded_bracket),     plot = TRUE)

Data: as.integer(test_dataset1$funded_bracket) in 6692 controls (as.integer(svm.pred) 0) < 1308 cases (as.integer(svm.pred) 1).
Area under the curve: 0.5062
```

After SMOTE Analysis
Decision Trees Output

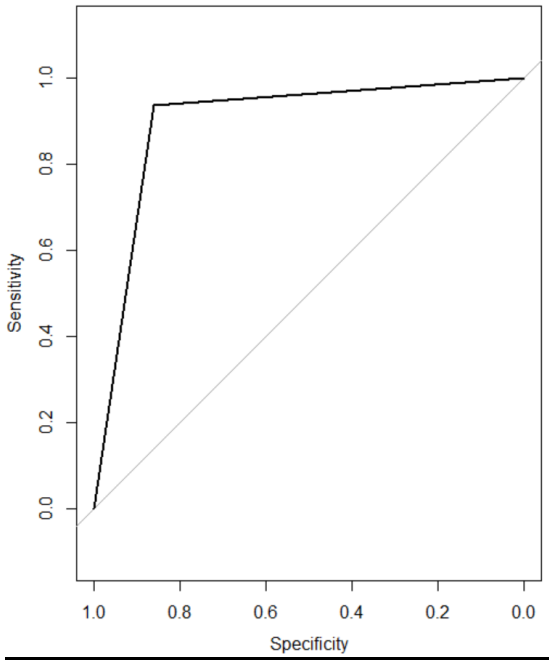
```
>   varImp(cvtree)
loess r-squared variable importance

  only 20 most important variables shown (out of 33)

                    Overall
H_req               100.000
term_in_months       76.756
requested_in_q1      68.801
L_req                55.134
ML_req               42.810
monthly_repayment    25.391
Irregular_repayment  25.199
male_borrower        17.214
female_borrower      17.214
sector_retail        12.771
requested_in_q2      12.343
sector_agriculture   11.418
requested_in_q3       9.044
purchase              3.816
buy                   3.636
sector_manufacturing  2.597
sector_arts           2.585
sector_housing        2.010
sector_services       1.309
sector_education      1.304
```






```
>   print(metrics)
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 524  84
         1  33 490

               Accuracy : 0.8966
                 95% CI : (0.8773, 0.9137)
    No Information Rate : 0.5075
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7933
 Mcnemar's Test P-Value : 3.791e-06

            Sensitivity : 0.9408
            Specificity : 0.8537
         Pos Pred Value : 0.8618
         Neg Pred Value : 0.9369
             Prevalence : 0.4925
         Detection Rate : 0.4633
   Detection Prevalence : 0.5376
      Balanced Accuracy : 0.8972

       'Positive' Class : 0
```

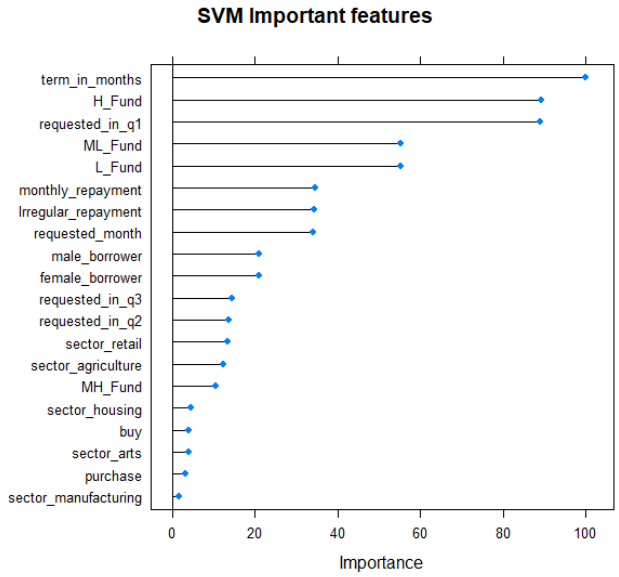
```
>   roc(pred.int, test_dataset$funded_bracket , plot=TRUE)

Call:
roc.default(response = pred.int, predictor = test_dataset$funded_bracket,     plot = TRUE)

Data: test_dataset$funded_bracket in 608 controls (pred.int 0) < 523 cases (pred.int 1).
Area under the curve: 0.8994
```

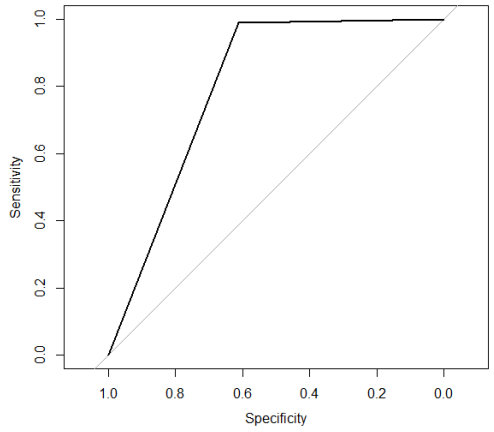


SVM

```
Call:
roc.default(response = as.integer(svm.pred), predictor = as.integer(test_dataset$funded_bracket),     plot = TRUE)

Data: as.integer(test_dataset$funded_bracket) in 942 controls (as.integer(svm.pred) 0) < 189 cases (as.integer(svm.pred) 1).
Area under the curve: 0.801
```

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 591 335
         1   0 205

               Accuracy : 0.7038
                 95% CI : (0.6762, 0.7303)
    No Information Rate : 0.5225
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.3901

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 1.0000
            Specificity : 0.3796
         Pos Pred Value : 0.6382
         Neg Pred Value : 1.0000
             Prevalence : 0.5225
         Detection Rate : 0.5225
   Detection Prevalence : 0.8187
      Balanced Accuracy : 0.6898

       'Positive' Class : 0
```

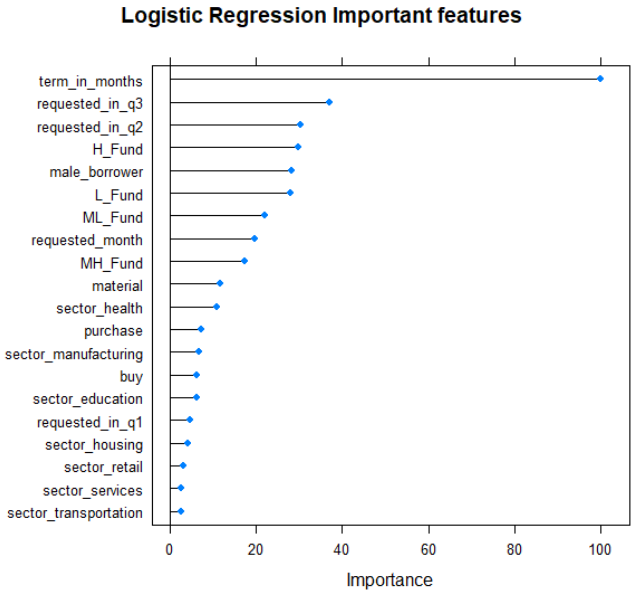




Logistic Regression

```
Call:
roc.default(response = as.numeric(glm.pred), predictor = as.numeric(test_dataset$funded_bracket,     plot = TRUE))

Data: as.numeric(test_dataset$funded_bracket, plot = TRUE) in 595 controls (as.numeric(glm.pred) 1) < 536 cases (as.numeric(glm.pred) 2).
Area under the curve: 0.8961
```

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 528  67
         1  51 485

               Accuracy : 0.8957
                 95% CI : (0.8764, 0.9129)
    No Information Rate : 0.5119
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.7911

 Mcnemar's Test P-Value : 0.1673

            Sensitivity : 0.9119
            Specificity : 0.8786
         Pos Pred Value : 0.8874
         Neg Pred Value : 0.9049
             Prevalence : 0.5119
         Detection Rate : 0.4668
   Detection Prevalence : 0.5261
      Balanced Accuracy : 0.8953

       'Positive' Class : 0
```



>

Naive Bayes

```
Call:
roc.default(response = pred.int, predictor = as.integer(test_dataset$funded_bracket),     levels = base::levels(as.factor(pred)), plot = TRUE)

Data: as.integer(test_dataset$funded_bracket) in 596 controls (pred.int 0) < 535 cases (pred.int 1).
Area under the curve: 0.9166
```

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0   0   0
         1 591 540

               Accuracy : 0.4775
                 95% CI : (0.448, 0.507)
    No Information Rate : 0.5225
    P-Value [Acc > NIR] : 0.9989

                  Kappa : 0

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.0000
            Specificity : 1.0000
         Pos Pred Value :    NaN
         Neg Pred Value : 0.4775
             Prevalence : 0.5225
         Detection Rate : 0.0000
   Detection Prevalence : 0.0000
      Balanced Accuracy : 0.5000

       'Positive' Class : 0
```