# **Contents**

**Abstract**

**Team Contributions**

**Problem Statement**

**Challenges Faced**

**Our Proposed Solution**

**Experiments Performed**

**Conclusion**

**Report on solution of ‘Give Me Some Credit’ Kaggle challenge**

# **Abstract**

In this report we provide a novel solution to the ‘Give Me Some Credit’ Kaggle challenge. We ensemble output from four classification algorithms into a regression model to evaluate the final predictions. The four classification algorithms used are Random Forest Classifier, Gradient Boosting Classifier, Ada Boost Classifier and Extra Trees Classifier. Upon submitting results, we obtained a private score of 0.866803 as against best score of 0.869558 in private leaderboard. We obtained a rank of 128 out of 925 teams – around **top 13.8 percentile** of team ranking.

# **Team Contributions**

The main idea belongs to Vishwaraj Anand. Every member of the group chose different classification algorithms individually and performed experiments on them. Each member came up with best estimates of parameters for their chosen algorithm to maximize the accuracy. After individual experiments, we assembled individual models and generated the final model. Each member re-performed the final tests individually combining each other’s implementations to gain hands-on training with all the classification algorithms used and to verify the team results thus obtained.

# **Problem Statement**

This challenge is setup in a banking environment where we have to predict whether a new customer is likely to be a loan defaulter or not. We had to build a model to determine the target class for a dataset based on ten input variables. Mathematically, it means that given a set of ten variables, learn what should be the target class.

|  |  |
| --- | --- |
| **Data to be learnt** | **Outcome to be predicted** |
| Several input datasets like: {Outcome, {var1, var2, … ,var10}} | Several result datasets like: {var1, var2, … ,var10} |

Here, the target variable had only two values – zero or one. Hence, this was a classic case of binary classification. First, we had to generate a classification model from the given data and outcome. Then, we had to use the learnt model to classify the result datasets and generate our predictions.

# **Challenges Faced**

We witnessed three main challenges to solving this problem: Missing values in training data, presence of outliers, and lack of simple statistical model to the problem. We have explained each of these below:

**1: Missing values in training data** – Several values were missing in training as well as test set in the field of ‘MonthlyIncome’ and ‘NumberOfDependents’. These were represented as ‘NA’ values in the given input files.

For unexplained values of NumberOfDependents, we assumed it as zero. This was supported by the fact that there was an overwhelming majority of datasets with zero values in NumberOfDependents field. This fact is best represented by the histogram in Table 1.

|  |
| --- |
|  |
| Table 1: Frequency vs NumberOfDependents in training dataset |

For MonthlyIncome, we seeded missing values with the average value [~6670] of MonthlyIncome for the whole dataset. This was a simple choice given the wide range of MonthlyIncome. This is also evident from the histogram in Table 2.

|  |
| --- |
|  |
| Table 2: Frequency vs MonthlyIncome in training dataset |

**2: Presence of outliers** – We had to determine the outliers from the dataset and ignore them from the training dataset. Upon analysis of various input field, we found outliers in ‘Age’ with some datasets had zero value of age. This was clearly an outlier as it did not have a physical meaning. We simply ignored such datasets from our training set instead of modifying such outlier datasets because of three reasons: first, such rows were very rare; second, we did not have such outlier rows in the test set; third, age field has a near Poisson distribution over Age and hence replacement with one value would have less confidence. This can be seen by the histogram in Table 3.

**3: Lack of simple statistical model to the problem** – We performed a multivariate linear regression over all variables provided, but achieved a poor accuracy, which is evident from the R-Score of a mere 5% and Kaggle score of 0.692303 as against best score of 0.869558 in private leaderboard with a rank in top 80%.

By doing regression, we realized that the simple regression models will not fit in to provide good scores, hence we had to determine our own solution for this problem.

|  |
| --- |
|  |
| Table 3: Frequency vs Age in training dataset |

# **Our Proposed Solution**

We built several prototypes with individual classification algorithms, but we will skip those development processes and instead provide our final solution directly. Our basic idea was to ensemble multiple classification algorithms through a regression model.

**Pre-processing:** We performed pre-processing to remove missing values over two fields: ‘MonthlyIncome’ and ‘NumberOfDependents’. For MonthlyIncome, because of many missing values, we decided to inject mean values in all of the missing values. For NumberOfDependents, we injected zero value in datasets with missing values. The explanatory reasons for both of these have been explained in above described ‘challenges with the problem’ section of the report.

We also performed pre-processing to remove outliers. We found that in Age field, certain datasets show zero age, but it did not appear as a plausible value given the bank setting of the problem where a person applying for bank loan must have had some age. Hence, we removed such training sets.

**Model used:** We built our model for final predictions in the following step by step approach:

1: we observed advantages and disadvantages of various classification algorithms: Random Forest Classifier, Gradient Boosting Classifier, Ada Boost Classifier and Extra Trees Classifier. And selected these four algorithms which complement each other’s pros and cons.

2: we achieved performances results from each of these individual ensemble algorithms over the number of estimators and tuned their parameters to achieve the best accuracy score from these individual algorithms.

3: we used cross-validation to combine the output of each algorithm and split the training set to 70%-30% dataset. Then we predict a regression model for outputs from all four models such that output from regression model is closest to the final goal.

4: we use output generated from individual algorithms and combined in the regression model for complete test dataset.

|  |
| --- |
|  |
| Figure 1: Our proposed prediction model for ‘Give Me Some Credit’ |

5: we fixed some outcomes which exceeded probability range of [0.0, 1.0] to their nearest boundary and generated the final answers.

Figure 1 represents a graphical representation for our proposed model, which is described above.

**Tools used:** We performed our experiments over Ms Excel in Windows and Python over Ubuntu 14.04.

In Ms Excel, we were able to perform the regression model and boundary checks on predicted value before submission.

In Python, we were able to tune our classification algorithms over the number of estimator trees. We used following libraries over python in Ubuntu 14.04 with 4GB ram and 4 CPU:

1: pandas – Handling datasets from input/output files

2: numpy – Handling large array operations

3: sklearn – Performing cross-validation of data within the training set

4: sklearn.metrics – Generating performance report for each classification model

5: sklearn.ensemble – Implementing the four independent algorithms showcased in this solution

# **Experiments Performed**

We performed many experiments to learn the best parameters in individual classification algorithms. We made several rounds of experimentation to understand how many number of tree estimators is required for achieving a good accuracy. Because all four of our individual classification algorithm at the bottom of our model [Figure 1] is based on ensemble methods, we needed to understand the correct set of parameters to train them individually.

**Round 1:** **Random Forest Classification**

After several rounds of evaluation, we found that Random Forest classification algorithm gives best results for 1000 decision trees. We observed that while the accuracy increases till 1000 decision trees, it did not increase significantly after that. At best, with an individual classification model, we achieved a private score of 0.851696, ranking 643 out of 925 teams.

**Round 2:** **Gradient Boosting Classification**

After several rounds of evaluation, we found that Gradient Boosting Classification algorithm gives best results for 1000 decision trees. We observed that while the accuracy increases till 1000 decision trees, it did not increase significantly after that. At best, with an individual classification model, we achieved a private score of 0.865291, ranking 226 out of 925 teams.

**Round 3: Ada Boost Classification**

After several rounds of evaluation, we found that Ada Boost Classification algorithm gives best results for 1000 decision trees. We observed that while the accuracy increases till 1000 decision trees, it did not increase significantly after that. At best, with an individual classification model, we achieved a private score of 0.860845, ranking 520 out of 925 teams.

**Round 4:** **Extra Trees Classification**

After several rounds of evaluation, we found that Extra Trees Classification algorithm gives best results for 600 decision trees. We observed that while the accuracy increases till 600 decision trees, it did not increase significantly after that. At best, with an individual classification model, we achieved a private score of 0.865291, ranking 709 out of 925 teams.

**Round 5: Using regression to make an ensemble of different classification models**

After all models were optimized, we made a regression out of these models through cross-validation by having a 70%-30% split of training dataset. Once, a regression model was made, we merged the individual output from each model and generated single value for each test dataset. Next, we performed boundary checks for the final probability and if probability exceeds [0.0, 1.0] range, we merge it to the nearest boundary.

By following these experiments, we were able to generate our final output and achieved a private score of 0.866803 as against the best score of 0.869558 in private leaderboard, which proved enough to break into **128 out of 925 teams [top 14% teams]**.

# **Conclusion**

We reported our novel solution for ‘Give Me Some Credit’ challenge of Kaggle where we used a regression model to ensemble four different classification algorithms to make predictions. We also showcased the various experiments we performed and the reasoning behind them. Finally, we showcased, how we achieved an overall rank of 128 out of 925 listed teams. [Top 13.8 percentile]