IMPROVING CREDIT RISK PREDICTION: BUILDING A SEMI-SUPERVISED LEARNING FRAMEWORK

1. Project overview

Description: This project aims to explore the use of semi-supervised learning to improve credit risk prediction

Problem statement:

Supervised machine learning models rely on labeled data to make predictions. In African financial markets, labeled data is often scarce and hard to obtain. Assigning labels to this data is expensive according to Chawla and Karakoulas ­­[1]. This limits the scope of credit risk prediction using supervised machine learning models in such markets. Our project aims to explore the use of semi-supervised learning techniques namely self-training and pseudo labelling to overcome the unlabeled data challenge and improve credit risk prediction for higher education students using supervised learning models. We aim to answer the question, “can semi-supervised learning improve credit risk prediction?”.

Objective:

To apply self-training and pseudo labeling on unlabeled data to achieve supervised classification model accuracy above 80%.

1. Dataset overview

Size and format:

The data is a 255 kilo byte comma separated value file.

Sources:

* Higher Education Students Loans and Grants Board (HESLGB)

Description:

The dataset has twelve thousand records and ten features including the target feature.

The format of the data is comma separated. It is unbalanced in terms of loan repayment status whereby more students have either fully or partially repaid their loan.

Features:

The dataset the following features as some of the variables; gender, marital status, age range, informed consent, convene of repayment process, highest education, program of study and region.

|  |  |  |
| --- | --- | --- |
| Feature | Type | Description |
| BAge | int64 | Borrower's age in years. |
| BMaritalStatus | Object | Borrower's marital status (e.g., Married, Single, Divorced). |
| BEduLevel | Object | Borrower's education level (e.g., Undergraduate, Postgraduate). |
| Householdsize | float64 | Number of members in the borrower's household. |
| Dependents | float64 | Number of dependents the borrower supports financially. |
| BStateR | int64 | Borrower's region/state code. |
| BTimeToComp | int64 | Time (in years) the borrower took to complete their education. |
| BMajor | Object | Borrower's academic major (e.g., Bachelor of Science in Midwifery). |
| BCollege | Object | Name of the institution the borrower attended. |
| CollegeYrOfEntry | int64 | Year the borrower entered college. |
| BGradYear | int64 | Year the borrower graduated. |
| OntimeGraduation | Object | Indicates whether the borrower graduated on time (e.g., Yes, No). |
| BEmpStatus | Object | Borrower's employment status (e.g., Employed, Unemployed, Self-employed). |
| RegionUnemploymentRate | float64 | Unemployment rate in the borrower's region at the time of loan application. |
| GradRate | float64 | Graduation rate of the borrower's institution. |
| CurrentLintRate | float64 | Current loan interest rate applicable to the borrower. |
| OgLIntRate | float64 | Original loan interest rate at the time of disbursement. |
| LTerm | int64 | Loan term in years. |
| Loangraceperiod | int64 | Grace period (in years) before loan repayment begins. |
| Repayment\_status | String | Loan repayment status |
| LPaymentDate | Object | Date of the borrower's last loan payment. |
| Repaymentplan | Object | Borrower's repayment plan type (e.g., Salary, Voluntary). |
| RTime | int64 | Time (in months) borrower has been repaying the loan. |
| ColEfforts | Object | Collection efforts made by the lender (e.g., Notices, Legalities). |
| LAmount | int64 | Loan amount originally disbursed. |
| FPaymentDate | Object | Date of the borrower's first loan payment. |
| RAmnt | int64 | Total repayment amount made by the borrower to date. |
| DeliqDate | Object | Date of loan delinquency occurrence, if applicable. |
| DeliqStatus | int64 | Status of loan delinquency (1 = Delinquent, 0 = Not Delinquent). |
| lossDefault | int64 | Loss given default as a percentage. |

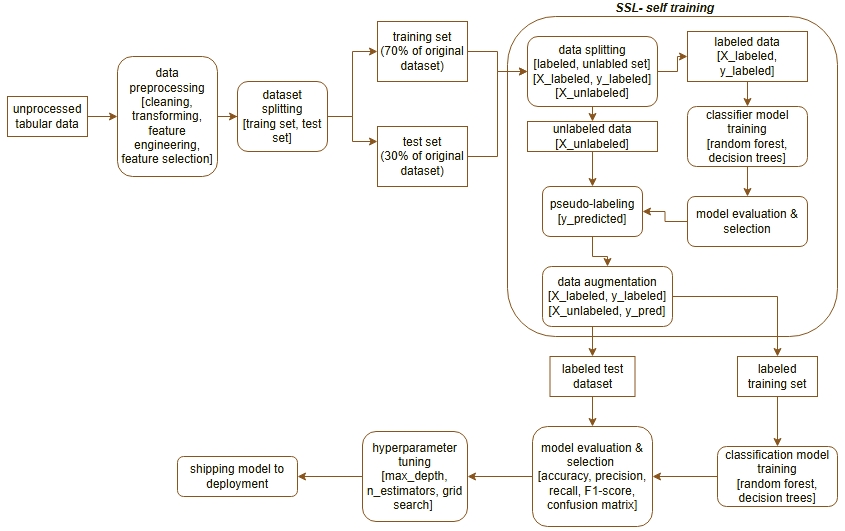
Data issues:

* Missing values in FPymentDate, BEmStatus
* Class imbalance in Repayment\_status
* Dataset size too small for model training
* Bad formatting of column names and values

1. Data preprocessing and feature engineering

|  |  |  |
| --- | --- | --- |
| Step | Method | Justification |
| Handling missing values | Mode imputation in FPaymentDate  Removing columns based on a 50% threshold of missing values | Improve model predictive accuracy |
| Formatting | Converting all column names to lower case including their values | To allow easy variable manipulation and avoid mistakes due to typing errors |
| Oversampling | Use borderlineSMOTE to mitigate the class imbalance | Improve model performance |
| Feature creation | Create age\_group from age range | Explicitly show to which group a loan applicant belongs to |
| Created feature probability of default | Replace reapayment status values (fully\_paid: low, partially\_paid: medium, unpaid: high) | Clear defined target feature |
| Created feature loss given default | LGD = EAD \* (1 - Recovery Rate) | Good predictor for credit risk |
| Encoding | Label encoding for target variable, one hot encoding for marital status, target encoding for other categorical features | Ease model training and testing |
| Data splitting | 70 – 30 split into training and testing sets | To avoid data leakage after semi-supervised learning |

1. Data pipeline design



The pipeline will be implemented using python functions.

Steps in the pipeline

1. Load csv data file
2. Perform data preprocessing (handle missing values: dropping based on threshold of 50%, impute using mode)
3. Feature engineering
4. Feature transformation
5. Dataset splitting (into training set and testing set using the 70 – 30 split)
6. Applying semi-supervised learning for both training and testing sets separately
   1. Data splitting (into labeled and unlabeled sets)
   2. Train a classifier model using the labeled set (random forest classifier)
   3. Use the classifier model to predict target labels
   4. Augment data. Combine the initially labeled set and the latter labeled sets
7. Use the training set to train a classifier model (ensemble models, decision trees)
8. Use the test set to evaluate and validate the models
9. Select best performing model and use it to predict credit risk
10. Model design and selection

Modes considered:

1. Decision trees
2. Random forest classifier
3. Artificial neural networks

Artificial neural networks were dropped from our proposed models of choice due to its computational complexity and implementation time.

Cross validation strategy:

Hyperparameter tuning:

ROC / AUC:

Selected model:

Random forest classifier

Justification: Better performance in terms of accuracy compared to decision trees.

Hyperparameters: n\_estimators = 120, max\_depth = 15

Evaluation metrics: Accuracy, Precision, F1 score, Recall, ROC, AUC

Model deployment plan: The final model will be saved into a joblib file and be incorporated using python’s fastAPI.

1. Tools and technologies

|  |  |
| --- | --- |
| Category | Tool/library |
| Programming language | Python 3.11 |
| Libraries:  Visualization  Data preprocessing  Modeling  Evaluation  Hyperparameter tuning  Data generation | Matplotlib, seaborn  Pandas, numpy  Scikit-learn  Scikit-learn  Scikit-learn  SMOTE, sdv |
| IDE | Jupyter notebook, visual studio code |
| Version control | Git, GitHub |
| Storage | Local storage, google drive, csv files |
| Saving models and pipelines | joblib |

1. Risks and mitigation

|  |  |
| --- | --- |
| Risk | Mitigation |
| Small dataset size | Synthetic data generation |
| Class imbalance | Oversampling |
| Privacy concerns | Using data stripped of names and identification |
| Categorical data | Limited scope of solutions to handle categories properly |

1. Timeline and milestones

|  |  |  |  |
| --- | --- | --- | --- |
| Task | Start date | End date | Status |
| Data collection |  |  |  |
| Data cleaning |  |  |  |
| Feature engineering |  |  |  |
| Feature selection |  |  |  |
| Synthetic data generation |  |  |  |
| Semi-supervised learning |  |  |  |
| Model training |  |  |  |
| Design document submission |  |  |  |

1. Appendices

Appendix A: confusion matrix

Appendix B: data cleaning (dropping missing values based on threshold 50%)

Glossary:

* SSL: semi-supervised learning, a technique of making use of a small amount of labeled data and a larger amount of unlabeled data in modelling.

1. References